Predicting the Past: Mathematical Models and Numerical Methods in Molecular Phylogenetics

Marnus Stoltz

a thesis submitted for the degree of

Doctor of Philosophy

at the University of Otago, Dunedin,

New Zealand.

31 March 2020

Abstract

Molecular phylogenetics is the study of phylogenies and processes of evolution by the analyses of DNA or amino acid sequence data. In this thesis we describe a computationally efficient Bayesian methodology for inferring species trees and demographics from unlinked binary markers. The new diffusion approach coupled with state-of-the-art numerical algorithms allow for analyses of datasets containing hundreds or even thousands of individuals. We demonstrate the scale of analyses possible using a SNP data sampled from 399 fresh water turtles in 41 populations. The method, which we call SNAPPER, is the successor of the coalescent based method SNAPP. A reanalysis of soybean SNP data demonstrates that the two methods are hard to distinguish in practise. We also describe a Bayesian methodology for inferring niches of present and ancestral species of plants from environmental measurements and estimated phylogenies. Fitting the phylogenetic niche model to three conifer species endemic to New Zealand confirms that viable ancestral niches can be inferred. Lastly, in anticipation of even larger genomic datasets we look into graphical processing units as computational tools for efficient model fitting. We introduce a new graphical processing unit based algorithm designed to fit long chain Hidden Markov models, applying this approach to an Hidden Markov model for nonvolcanic tremor events. Our implementation resulted in a 1000-fold increase in speed over

the standard single processing algorithm, allowing for a full Bayesian inference of model parameters.

Acknowledgements

First and foremost, I would like to thank my supervisor David Bryant for patiently guiding me through the imaginarium of logic and reason: The clarity you bring and enjoyment you carry when tackling hard problems are genuinely inspiring!

Ek wil ook dankie sê aan my gesin: Ma en Pa, sonder julle liefde en ondersteuning sou ons nie lewe voluit kon aanpak nie (Ma dankie vir al die reelings en worry vir ons part; Pa dankie vir al die whiskey). Gene en Juan, sonder julle sou lewe aanpak. New Zealand gesin (Fanie, Sebbo en Suzelle) dankie vir julle gesellige huis en ope arms; New Zealand sou baie vêr voel andersins.

Thank you to all my collaborators, without whom this body of work would not have been possible. Especially my brother, whose visit added something special.

A shout-out to the residents of 32: Flatmate acquaintance Helen and brofist Chris thank you for instilling a sense of belonging (and lots of double wiggles). It was a laugh!

I would also like to highlight some miscellaneous things that contributed to the experience (rather than the quality of work) in some way:

- The music of GoGo Penguin, Lambert and Boards of Canada (courtesy of David) for accompaniment to the long grind sessions;
- All the sci-fi books for injecting a daily dose of escapism;
- My running shoes for the life lessons;
- The blue Suub for providing a reliable way to explore the pristine beauty of New Zealand;

- The old man who lost my passport and the old sensei who paints mountains with a shovel;
- The Irishman, in traditional dress, who got in an argument one evening at dinner with the captain of the Queen Mary II; During which he unexpectedly outmanoeuvred the knight of the British empire by standing on his chair and subsequently falling off it, laying bare the truth;
- The infinite sake bar (may I never set foot in it again);
- The futon on which I regularly took naps;
- The desk under which I regularly took naps;
- The Wacaco espresso hand pump for keeping me from taking naps too regularly;
- Maddy's chai for its warmth and comfort;
- The laser experiments that kept me awake at night;
- Bitcoin arbitrage for helping me pay the rent.

Laastens, aan al my pelle wat vêr was: Julle ou poespasse word altyd gemis. Mag ons saam grapgatte wees tot die bitter einde toe...

Contributions

The writing of Chapter 1 was done by me with editing input from David Bryant.

Most of Chapter 2 and parts of Chapter 3 was submitted to the journal Systematic Biology in a manuscript entitled: "Bayesian inference of speceies trees using diffusion models". Most of Chapter 2 was jointly written by David Bryant and me. Boris Baeumer and Colin Fox assisted with theoretical questions behind diffusion models. Remco Bouckart implemented the diffusion model in Beast2 based on a Python prototype coded by me. Numerical experiments were conducted by me with design input from David Bryant. Analysis of data was conducted by me with genomic data of freshwater turtles provided by Arthur Georges. Chapter 3 is written by me partly assisted by David Bryant. Numerical experiments were conducted by me with design input from Colin Fox and David Bryant. This work develops previous work done by Gordon Hiscott.

Chapter 4 was written by me with editing input from David Bryant. Models and experiments was set up by me with help from David Bryant and Colin Fox. Conifer data and analysis platform was provided by Steve Higgins and Matt Larcombe. This chapter extends previous work done by Steve Higgins.

Chapter 5 was submitted to the Journal of Computational and Graphical Statistics in a manuscript entitled: "A 1000-fold Acceleration of Hidden Markov Model Fitting using Graphical Processing Units, with application to Nonvolcanic Tremor Classification". The writing was done jointly by David Bryant, Gene Stoltz and me with editing input from Ting Wang. The model was implemented and numerical experiments were conducted

jointly by Gene Stoltz and me. The data analysis was done by me with tremor data provided by Kazushige Obara.

Contents

List of Tables

List of Figures

List of Algorithms

List of Symbols

$\approx$	Approximately equal to
$\cap$	Intersection of two sets
$\cup$	Union of two sets
Δ	A small interval of
$\delta(.)$	A dirac delta function
ϵ	A small amount
$\in$	An element of
$\langle ., . \rangle$	The real innerproduct
$\mathbb{C}$	Set of complex numbers
$\mathbb{N}$	Set of natural numbers
$\mathbb{R}$	Set of real numbers
$\mathcal{A}_t$	Continuous Markov chain backward in time
$\mathcal{A}_t^N$	Discrete Markov chain backward in time
$\mathcal{C}^k_{[0,1]}$	Space of continuous functions with k derivatives on $[0, 1]$
$\mathcal{O}(.)$	Order of
$\notin$	Not an element of
Ω	A domain
$\otimes$	The Kronecker product of two matrices
$\prod$	The product of
$\rightarrow$	Tends toward in the limit
$\sim$	Approximate amount of time it took
$\sum$	The sum of

$\sum^{'}$	The sum with half of the first element
$\{.\}$	A set consisting of
I	State space of a continuous Markov process
I_N	State space of discrete Markov process, where N indicate number of states
$Pr[.]$	A value between 0 and 1 indicating a probability
X_t	Continuous Markov chain forward in time
X_t^N	Discrete Markov chain forward in time, where N indicate number of states
vec(.)	The vectorization of a matrix

Chapter 1

Basic models and key concepts

1.1 Thesis blueprint

The work in this thesis consists of three distinct but similar parts, namely: Chapter 2 and Chapter 3; Chapter 4 and Chapter 5. (Note that Chapter 1 is excluded since it serves as a review for key concepts that we encounter throughout the rest of the thesis). Each part consists of the same four key components:

- A mathematical model.
- A concrete dataset for fitting the mathematical model.
- A Bayesian inference framework for quantifying uncertainty around model parameters.
- Numerical and computational methods for calculating model likelihoods of model parameters given the data.

However, the core of the thesis is Chapter 2 and Chapter 3. The remaining two chapters two chapters, Chapter 4 and Chapter 5, either extend or feed into one of the above mentioned components of the thesis core. We give more details below.

In the core chapters we study the application of mathematical models to infer species trees and related evolutionary processes given unlinked binary markers. We explore the interplay between models and establish a solid theoretical underpinning for diffusion models in particular. We present a linearithmic time algorithm to solve diffusions on species trees. This algorithm forms the core of a new Bayesian software package for phylogenetic analysis. We test the software extensively and showcase its capability. The approximation framework for diffusion models is introduced and we

carefully derive the numerical results. Our approach relieves some of the computational strain that available methods are experiencing in the advent of large genomic datasets.

In Chapter 4 we extend statistical inference of the core chapters in some sense. More specifically, we introduce phylogenies (possibly inferred using methods developed in the core chapters) to well-established niche models for plants. In doing so we allow for joint inference of not only present-day species niches but also niches of common ancestral populations. We test the robustness of the model and give proof-of-concept results.

There exist opportunity to exploit parallel data structures associated with models in phylogenetics. However numerical algorithms developed in this thesis can be tricky to implement directly on specialised computational hardware such as GPUs without any prior experience. Therefore in Chapter 5 we use a concrete example (with application in seismology) to develop a better understanding of the underlying design principles for implementing numerical algorithms on the computing architecture of GPUs. In the process we implemented a parallel algorithm for computing likelihoods of hidden Markov models. Our implementation resulted in a 1000-fold increase in speed over the standard single processor algorithm. It forms part of a Bayesian software package for seismological analysis.

1.2 Discrete models for gene frequencies

1.2.1 The Wright-Fisher model

The Wright-Fisher model (Fisher, 1930; Wright, 1931) is one of the cornerstones of theoretical population genetics. In its simplest form the model describes the transmission of genes in a population along generations using a random reproduction mechanism. It has many variations and extensions, and has had a massive impact across population genetics as well as phylogenetics (Ewens, 2004; Felsenstein, 2004). For this work our starting point is the Wright-Fisher discrete model of drift and mutation. The model makes the following explicit assumptions (Hein *et al.*, 2004):

- Generations are discrete and non-overlapping;
- Individuals are assumed to be haploid (the population size is doubled in case it is a diploid population);
- There is no differences in selective advantage between allele types;

- There is no geographical or social structure in the population;
- No recombination takes place within the gene sequences.

We will assume two alleles types, which we denote 'red' and 'green'. Given these assumptions we model the number of red alleles observed at a locus for a population of size $2N$ as a Markov chain

$$X_t^{2N} \qquad \text{for } t = 0, 1, 2, \ldots,$$

with the red allele frequency represented in the state space

$$I_{2N} = \left\{ \frac{i}{N} : i = 0, 1, 2, \ldots, 2N \right\}.$$

Transition probabilities are given by the binomial distribution

$$Pr\left[X_{t+1}^{2N} = j \mid X_t^{2N} = i\right] = \binom{2N}{j} p_i^j (1 - p_i)^{2N-j}, \tag{1.1}$$

where

$$p_i = (1 - u)\frac{i}{2N} + v\left(1 - \frac{i}{2N}\right) \tag{1.2}$$

describes the probability of drawing a red allele in generation $t = 0, 1, 2, \ldots$. The mutation rate from red alleles to green alleles (forward mutation) is u and the mutation rate from green alleles to red alleles (backward mutation) is v. We assume neutral mutations. In Figure 1.1 we illustrate a random sample path of X_t^5 for generations $t = 0, 1, 2, 3$. Each generation is made up of individuals randomly drawn with replacement from the previous generation with a chance of mutating.

1.2.2 The Moran model

The Wright-Fisher model can be tough to work with due to the complicated nature of the transition probabilities. The Moran model (Moran, 1958) has a much simpler reproduction scheme. It can be informally described as follows: For every step draw two individuals with replacement from a population of size $2N$, randomly choose one individual to reproduce an offspring and one individual that dies off. Replace the individual that died with the offspring of the individual that reproduced. The offspring

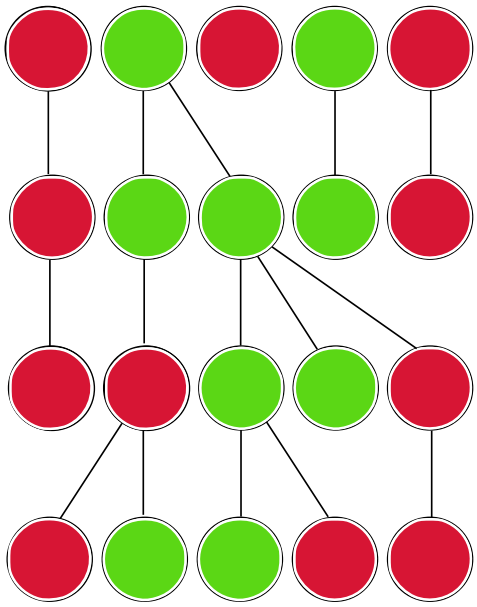


Figure 1.1: The discrete Wright-Fisher model illustrated for 4 generations (t) of a haploid population of 5 individuals, starting with the initial population at the top. Each consecutive generation ($k = 1, 2, 3$) is made up of individuals randomly drawn with replacement from the previous generation with a chance of mutating.

in the replacement step has a probability of mutating. Note that under this model, generations are overlapping; not every step corresponds to a distinct generation.

The red alleles observed in a population can be described as a Markov chain

$$X_t^{2N} \qquad \text{for } t = 0, 1, 2, \ldots,$$

with the red allele frequency represented in the state space

$$I_{2N} = \left\{ \frac{i}{2N} : i = 0, 1, 2, \ldots, 2N \right\}$$

and transition probabilities

$$Pr\left[X_{t+1} = i - 1 | X_t = i\right] = \frac{(i)(2N - i)(1 - v) + ui^2}{2N^2} \tag{1.3}$$

$$Pr\left[X_{t+1} = i + 1 | X_t = i\right] = \frac{(i)(2N - i)(1 - u) + v(2N - i)^2}{2N^2} \tag{1.4}$$

$$Pr\left[X_{t+1} = i | X_t = i\right] = 1 - \sum_{j=i,i+1} Pr\left[X_{t+1} = j | X_t = i\right]. \tag{1.5}$$

In Figure 1.2 we illustrate a random sample path of X_t^5 for timesteps $t = 0, 1, 2, 3$. Here $2N$ timesteps are considered to be one generation unlike the Wright-Fisher model

where one timestep equals one generation.

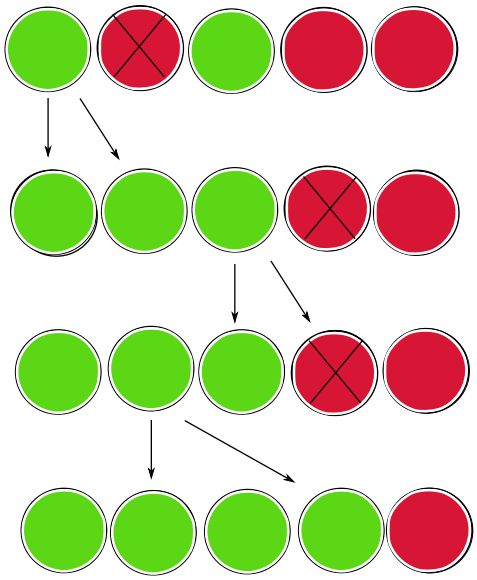


Figure 1.2: The Moran model illustrated for 4 timesteps (not generations) of a haploid population of 5 individuals, starting with the initial population at the top. Each timestep ($t = 1, 2, 3$) consists of randomly selecting one individual to produce (with a chance of mutating) an offspring and one individual to be replaced by the offspring.

1.3 Diffusion models

1.3.1 Approximating discrete random walks

Diffusion models make it viable to model allele frequencies efficiently. The idea behind diffusion models is to approximate the normalized discrete random walk ($\frac{1}{2N}X_t^{2N}$) by a continuous random walk (X_t) with state space $[0, 1]$ that is easier to work with analytically. Instead of considering discrete generations, we construct a random walk which is continuous in time. We set up the approximation so that the larger N gets, the better the approximation fits.

Modelling gene frequencies this way involves a rescaling of time. There are simple, practical, reasons for this. As the population size N gets larger, the rate of genetic drift decreases, ultimately approaching zero drift in the limit. However the effect of drift is something that we would like to model. For this reason we change the units of time so that the rate of drift remains approximately the same as N increases, eventually converging to some non-zero amount. The convention for diploid populations is to use a scale where 1 unit of time corresponds to $2N$ generations (one coalescent unit).

If we are to change the units of time, we need to adjust the rate of mutation accordingly. Therefore the overall rate of change due to mutations from green alleles to red alleles becomes $2Nu$ and the overall rate of change due to mutation from red alleles to green alleles becomes $2Nv$. We adopt the standard notation and define $\beta_1 = 2Nu$ and $\beta_2 = 2Nv$.

It is not trivial to show that the discrete random walk X_t^{2N} converges to a continuous random walk X_t when N becomes large. Deep theory and intricate mathematical constructs help deal with the issue of establishing convergence of a discrete random walk. See Ethier and Kurtz (1986).

There is a set of conditions (Ethier and Kurtz, 1986, Theorem 2.11, p. 172) that, if satisfied, guarantees convergence for a discrete random walk. These conditions are related to how much the discrete random walk (in our case allele frequency) can change per timestep (generation). In practise if we can show that this change is well-behaved then we have convergence. In fact if we can derive this change explicitly we can also uniquely define the continuous random walk X_t. Two quite different discrete processes can converge to the same continuous process. For example, the discrete random walk under the Moran model and the discrete random walk under the Wright-Fisher model are seemingly very different and yet they converge to the exact same continuous random walk X_t (the only difference is the timescale). These derivations for the Moran model and Wright-Fisher model are standard and we refer to Etheridge (2011) for details.

The continuous random walk X_t is an example of a *diffusion process* and is most easiest defined in terms of *partial differential equations* (PDEs) describing the relationship between a multivariate function and its derivatives. The specific class of time-dependent PDEs we will use are called *parabolic partial differential equations*. Below we discuss two important related parabolic PDEs that uniquely define the diffusion process X_t that approximates the discrete random walk under the Moran model as well as the discrete random walk under the Wright-Fisher model.

1.3.2 The forward diffusion

Firstly, for each value $t > 0$ of the diffusion process X_t we let $f(x,t)$ denote the *density* of the allele frequency x at time t, where x is defined on the state space $[0,1]$. Surprisingly, there is very little choice over what the function $f(x,t)$ might be, after a few basic assumptions are made. (Though see McKane and Waxman (2007) for a discussion about how we need to augment $f(x,t)$ with point masses at 0 and 1.)

As is often the case in mathematical modelling (and as mentioned above), we work

with the function f indirectly using a partial differential equation (PDE). Stochastic process theory (Øksendal, 2003, chap. 8, pg. 161) tells us that the function f satisfies the PDE

$$\frac{\partial f(x,t)}{\partial t} = -\frac{\partial}{\partial x}\left(\beta_2(1-x) - \beta_1 x\right) f(x,t) + \frac{1}{2}\frac{\partial^2}{\partial x^2} x(1-x) f(x,t). \tag{1.6}$$

The PDE (1.6) is an example of a Kolmogorov Forward equation or Focker-Planck equation. We refer to the PDE (1.6) simply as the *forward diffusion*.

The forward diffusion by itself is not enough to uniquely determine f. We also need to specify what f looks like at the boundaries. To specify that the distribution of the initial state is given by some density π we add the initial condition

$$f(x,0) = \pi(x) \text{ for } x \in [0,1]. \tag{1.7}$$

Note that to fix the initial state at a specific value x_0, we need π to be a Dirac-delta function $\pi(x) = \delta(x - x_0)$ which is essentially a infinitely thin spike at x_0. Even with these initial conditions, the PDE (1.6) does not uniquely determine the function f. We also need to add boundary conditions at $x = 0$ and $x = 1$ to guarantee that the probability of going outside the interval $[0, 1]$ is always zero. In the appendix we show that specifying a condition on the integral of f and then plugging in the forward diffusion, eventually leads to what in physics is known as a *zero-flux condition*

$$-(\beta_2(1-x) - \beta_1 x) f(x,t) + \frac{1}{2}\frac{\partial}{\partial x} x(1-x) f(x,t) = 0, \tag{1.8}$$

when $x = 0$ or $x = 1$ (see also McKane and Waxman (2007)).

The diffusion approximation works well only if mutations rates are assumed small. As shown in Ethier and Norman (1977) we have convergence in expectation with an error bound on the diffusion approximation behaving like $O(u + v + 1/N)$. If u or v are large (compared to, say, $\frac{1}{N}$) then the diffusion approximation will fail miserably. Nevertheless, in a recent simulation study, Tataru *et al.* (2017) used simulations to quantify the error from the diffusion approximation in small populations, and found that diffusion models gave reasonable approximations even when the population has fewer than 100 individuals.

1.3.3 The backward diffusion

It is not just the density f of the diffusion process X_t that can be expressed in terms of a PDE. We can also express the expected value of allele frequency in the past conditioned on allele frequency in the present. More specifically, let X_T denote the allele frequency in the present and suppose h is an integrable function. Let $g(x,t)$ denote the expectation of $h(X_T)$ conditioned on allele frequency x at time $T-t$, where $0 < t < T$

$$g(x,t) = E(h(X_T)|X_{T-t} = x).$$

Then according to Øksendal (2003, chap. 8, pg. 133) the function g satisfies

$$\frac{\partial g(x,t)}{\partial t} = (\beta_2(1-x) - \beta_1 x)\frac{\partial}{\partial x}g(x,t) + \frac{1}{2}x(1-x)\frac{1}{2}\frac{\partial^2}{\partial x^2}g(x,t). \tag{1.9}$$

The PDE (1.9) is sometimes known as the Kolmogorov Backward equation, we will refer to it as the *backward diffusion*. In Chapter 2 we specify smooth (infinitely differentiable) functions π^* appearing in likelihood calculations we use as the function h.

There is an interplay between the differential operators (i.e the right hand side of a PDE) of the forward diffusion and backward diffusion. In mathematical terms they are called *adjoint operators* of one another. Deriving the backward diffusion from the forward diffusion (with zero-flux boundary condition) using the fact that we want the differential operators to be adjoint we find that in addition to (1.9),

$$x(1-x)\frac{\partial}{\partial x}g(x,t) = 0$$

must be satisfied at the boundary $x = 0$ and $x = 1$ for all $t > 0$.

Furthermore to ensure that our solution $g(x,t)$ is unique we also require the boundary condition

$$x(1-x)\frac{\partial^2}{\partial x^2}g(x,t) = 0$$

at $x = 0$ and $x = 1$. Both these conditions are automatically satisfied when $g(x,t)$ is smooth. It follows from the result in Epstein and Mazzeo (2010, pg. 595) that the solution $g(x,t)$ is smooth if the function h on the boundary is smooth. We give details in Appendix B.

1.4 The n-coalescent

1.4.1 Basic model

The n-coalescent (Kingman, 1982b) is a model for the distribution of gene trees for a sample from a single population. Coalescent theory, which is built on the n-coalescent, provides a powerful framework for inference of evolutionary processes using these gene trees (we discuss the most widely used extension of the coalescent below). Kingman (2000) points out the three insights that make up the core of the coalescent. The first insight is the idea of tracing the ancestry (lineage) of a gene backward in time and building up the gene tree (at a particular locus) in a population sample back to the point at which the sample individuals share a most recent common ancestor (MRCA). This is a generalization of 'identity by descent' (Nagylaki, 1989) to more than two genes. It becomes even more powerful because of the second insight, that for a large class of demographic models, characterized by selective neutrality and constant population size, the stochastic structure of the genealogy does not depend on the detail of the reproductive scheme. The third insight is that the mutation process is statistically independent of demographics when conditioned on the genealogy.

The *discrete-time n-coalescent* is the Markov chain

$$\mathcal{A}_t^{2N} \qquad \text{for } t = 0, 1, 2, \ldots$$

that describes the *number* of distinct ancestors (lineages) of n individuals in the present from a population of size $2N$. It has state space

$$I_n = \{i : i = 1, 2, \ldots, n\}.$$

Kingman (1982b) derived a useful approximation of this process under a general reproduction scheme (which includes the Wright-Fisher model and Moran model) for large populations by reducing the transition matrix of the Markov chain, writing it in terms of the probability that one coalescent event occurs in a generation

$$Pr\left[\mathcal{A}_{t+1}^{2N} = i - 1 \mid A_t^{2N} = i\right] = \binom{i}{2}(2N)^{-1} + \mathcal{O}(N^{-2})$$

and the probability that no coalescent event occurs in a generation

$$Pr\left[\mathcal{A}_{t+1}^{2N} = i \mid A_t^{2N} = i\right] = 1 - \binom{i}{2}(2N)^{-1} + \mathcal{O}(N^{-2}).$$

The probability of more than one coalescent event in a generation is shown to be $\mathcal{O}(N^{-2})$ and hence negligible for large N.

Rescaling time such that one unit of time corresponds to $2N$ generations, we approximate the waiting time until a coalescent event for a sample of lineages $i = 1, \ldots, n$ in a large population $2N$ by an exponentially distributed random variable with rate $\binom{i}{2}$. This implies (Kingman, 1982b) that as N become large $\mathcal{A}_{[2Nt]}^{2N}$ converges to a continuous-time Markov chain

$$\mathcal{A}_t \qquad \text{for } t \geq 0$$

with the same state space I_n and infinitesimal generator, in this case rate matrix, defined as

$$(q_{ij}) = \begin{cases} -\binom{i}{2} & \text{for } i = 1, \ldots, n; j = i \\ \binom{i}{2} & \text{for } i = 1, \ldots, n; j = i - 1 \\ 0 & \text{otherwise.} \end{cases}$$

This death process, starting at n and ending at 1, is what Kingman called the *n-coalescent*. A good overview of coalescent theory is given in Hein *et al.* (2004). We illustrate the discrete coalescent by tracing the lineage of a sample of three individuals given a general reproduction mechanism in Figure 1.3.

1.4.2 Model with mutation

One of the key advantages of the coalescent is that, for neutral mutations, the gene tree can be decoupled from the mutation process, a feature which forms the basis of many implementations. Briefly, we can model mutation events forward in time on top of the genealogy after the genealogy has been sampled. We start at the most recent common ancestor (MRCA) and evolve its state along the gene tree using a continuous Markov chain with state space ("red", "green") and a 2×2 rate matrix defined in terms of the forward and backward mutation rates. The number of mutation events on a branch are determined by standard pyhlogenetic models (Felsenstein, 2004). The n-coalescent

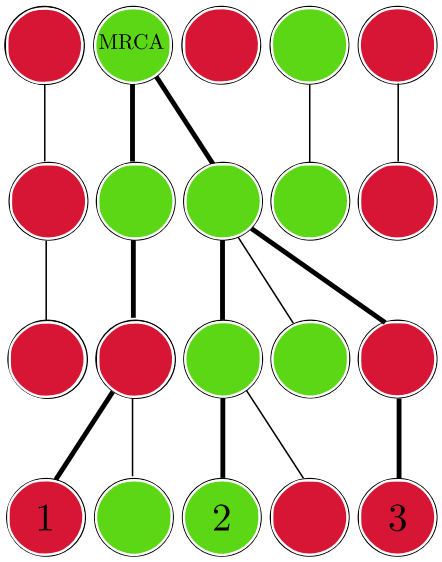


Figure 1.3: The genealogy of three randomly sampled sequences (i.e 1,2,3) in the present. The ancestry of the sequences are marked by bold lines four generations back in time until the most recent common ancestor is found (MRCA).

with forward and backward mutation can be used to derive a $\mathcal{O}(1/N)$ approximation for red allele densities that satisfies the backward diffusion (1.6), see Griffiths and Spano‘ (2010) for details and see Hein *et al.* (2004, Chapter 3) for variations on the basic coalescent that include population structure and selection.

1.4.3 The multispecies model

The multispecies coalescent is a model of the distribution of gene trees for gene samples from multiple species or populations. The idea is to restrict lineages so that they can only coalesce at some time if they are in the same ancestral population at that time. Note that a *speciation event* is an assumed point back in time when two populations are considered to have diverged.

The multispecies coalescent is currently the most widely used model in the systematics community for taking account of population dynamics when inferring species trees and other phylogenetic processes (e.g. Degnan and Rosenberg, 2009; Song, Liu, Edwards, and Wu, 2012; Bryant, Bouckaert, Felsenstein, Rosenberg, and RoyChoudhury, 2012; Rannala and Yang, 2017). For a general introduction to the multispecies coalescent see Felsenstein (2004). In Figure 1.4 we demonstrate the multispecies coalescent
.

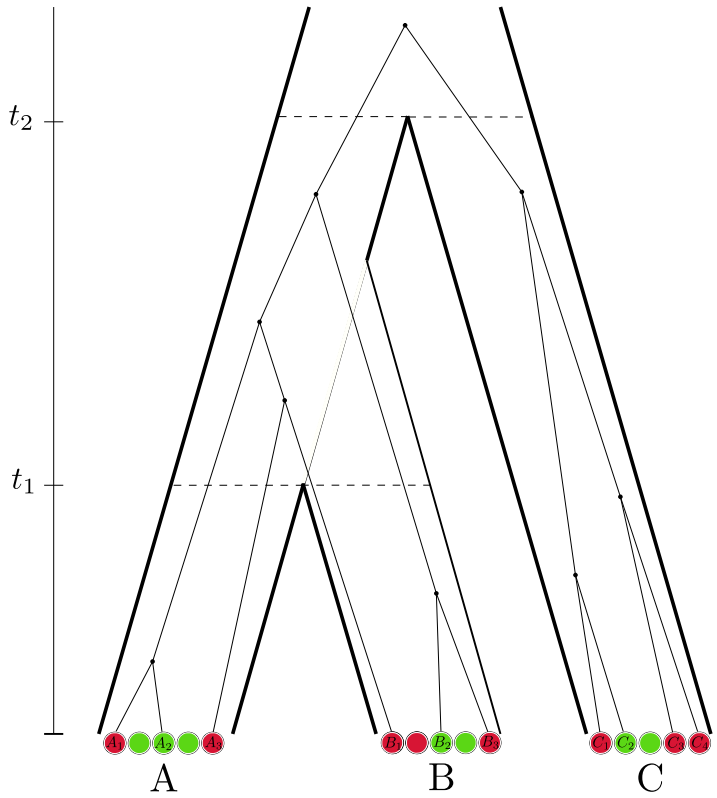


Figure 1.4: A simulated multispecies coalescent with three populations in the present (A,B,C). Speciation events in the past are indicated by t_1 and t_2. Note that here we assume the present to be at $t = 0$ with $0 < t_1 < t_2$ representing the amount of time that an event took place in the past. Samples can only coalesce if they are considered to be in the same ancestral population.

1.5 Bayesian inference

1.5.1 Overview

There are two important schools of statistical inference, Bayesian inference and frequentist inference. These two schools differ in the fundamental way in which they conceive uncertainty. In practise it seems that both approaches can contribute to statistical inference in some way (Bayarri and Berger, 2004; Raue *et al.*, 2013; Huelsenbeck and Rannala, 2004), however there are not so subtle differences in how inference results are presented and interpreted. This stems from how probability is defined differently in the frameworks of Bayesian- and frequentist inference. We refer to Press (2005) for a more in-depth discussion on this.

Typically, from a Bayesian point of view, probabilities are related to the knowledge of an event. The measured data is fixed and parameters are considered random quantities defined in terms of probability distributions and no true process or point measures exist.

In this thesis we use a Bayesian framework to quantify uncertainty around model parameters but we also compare fitted models using frequentist point estimates. Below we discuss some underlying theory of Bayesian inference and key methods for inferring and fitting models. We will use these methods throughout the model fitting sections in the chapters that follow.

1.5.2 Bayes' rule

Bayesian methods are based on a property of conditional probability known as Bayes' rule. Suppose θ is a set of parameters for a model of some process we are interested in. Furthermore suppose we are given data that represents measurements of the process and that we are interested in the distribution of parameters θ conditional on observing the given data, that is $\pi(\theta|\text{data})$ the *posterior distribution*. We specify a distribution $\pi(\theta)$ that codifies our knowledge and uncertainty about the parameters θ, independent of the data. We call $\pi(\theta)$ the *prior distribution*. Then we can adjust this hypothesis to better describe the data, by computing how likely observed data was generated by a given set of parameters θ. To do this we compute the *likelihood function* $\pi(\text{data}|\theta)$ and the *normalization constant* $\pi(\text{data})$. Properties of conditional probability give Bayes'

rule as

$$\pi(\theta|\text{data}) = \frac{\pi(\text{data}|\theta)\pi(\theta)}{\pi(\text{data})}.$$

In practice the normalization constant $\pi(\text{data})$ is difficult to compute due to the intractability of the integral

$$\pi(\text{data}) = \int \pi(\text{data}|\theta)\pi(\theta)d\theta.$$

Therefore computing the posterior distribution directly is intractable in most cases. Methods such as Markov Chain Monte Carlo (MCMC) allow us to generate samples from the posterior distribution and yet only require ratios of priors and likelihood functions, avoiding the need to determine $\pi(\text{data})$ explicitly.

1.5.3 Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm is a way to propose states asymmetrically when simulating an MCMC. The basic concept behind MCMC (e.g. Gelfand and Smith, 1990) is that a Markov chain can be constructed with a stationary distribution, equal to the target posterior distribution $\pi(\theta|\text{data})$. See Algorithm 1 for details.

It follows from Chib and Greenberg (1995) that the Markov chain $\{\theta_0, \ldots, \theta_N\}$ (simulated using Algorithm 1) eventually converges to the right density (target posterior distribution; provided the conditions are right). In practise when assessing convergence of a Markov chain $\{\theta_0, \ldots, \theta_N\}$ we want to diminish the effect of the starting distribution. Therefore early samples of the chain will be discarded, this is called *burn-in*.

The fraction of burn-in and length of the chain (N) are dependent on the efficiency of the MCMC algorithm (rate of convergence to target posterior distribution; number of independent samples) which in turn is critically reliant on the choice of proposal distribution q. A good deal of experimentation can be required in order to get a reasonably efficient MCMC.

Algorithm 1 Markov Chain Monte Carlo simulation

1: **procedure** MCMC($\pi(.)$,$\pi(.|\text{data})$, θ_0, q)
2: Draw θ_0 from the prior distribution $\pi(\theta)$
3: **for** each step $t = 1, \ldots, N$ **do**
4: Sample θ^* from proposal distribution $q(\theta|\theta_{t-1})$
5: Compute

$$\alpha(\theta^*, \theta_{t-1}) = \min\left(1, \frac{\pi(\theta^*|\text{data})}{\pi(\theta_{t-1}|\text{data})}\frac{q(\theta_{t-1}|\theta^*)}{q(\theta^*|\theta_{t-1})}\right)$$

6: Set

$$\theta_t = \begin{cases} \theta^* \text{ with probability } \alpha(\theta^*, \theta_{t-1}) \\ \theta_{t-1} \text{ with probability } 1 - \alpha(\theta^*, \theta_{t-1}) \end{cases}$$

7: **end for**
return $\{\theta_0, \ldots, \theta_N\}$
8: **end procedure**

Metropolis-Hastings algorithm (Gelman *et al.*, 2013, chap 11, pg.289, see Algorithm 1) requires a proposal distribution $q(\cdot|\theta)$ for each θ which we can sample from, and for which we can compute the ratios. The initial parameters θ_0 are typically drawn from the prior distribution $\pi(\theta)$. Using Bayes' rule

$$\pi(\theta|\text{data}) = \frac{\pi(\text{data}|\theta)\pi(\theta)}{\pi(\text{data})}$$

we can conveniently dispose of the normalization constant $\pi(\text{data})$ in the acceptance ratio $\alpha(\theta^*, \theta_{t-1})$ (see line 5)

$$\left(\frac{\pi(\text{data}|\theta)\pi(\theta)}{\pi(\text{data})}\right) \Big/ \left(\frac{\pi(\text{data}|\theta^*)\pi(\theta^*)}{\pi(\text{data})}\right) = \frac{\pi(\text{data}|\theta)\pi(\theta)}{\pi(\text{data}|\theta^*)\pi(\theta^*)}.$$

This trick allows us to simulate from an intractable target posterior distribution using only the likelihood and the prior.

Chapter 2

Diffusion models on species trees

2.1 Introduction

In this chapter we describe a new and computationally efficient Bayesian methodology for inferring species trees and demographics from unlinked binary markers. Likelihood calculations are carried out using diffusion models of allele frequency dynamics (which we discuss in Chapter 1) combined with a new algorithm for numerically computing continuous likelihoods. The diffusion approach allows for analysis of datasets containing hundreds or thousands of individuals. The method, which we call Snapper, has been implemented as part of the Beast2 package. We introduce the models, the efficient algorithm, and report performance of Snapper on simulated data sets and on SNP data from rattlesnakes and freshwater turtles. We give the details of the numerics in the next chapter.

Recent years have witnessed a proliferation in the number of methods for inferring species trees from whole genomes (Bryant and Hahn, 2020). Some have gone so far as to describe this as a paradigm shift in phylogenetics (Edwards, 2009). It is now widely accepted that (i) phylogenetic analysis needs to take account of the varying evolutionary histories of different parts of the genome, and (ii) estimation of evolutionary relationships between populations (or species) should take account of evolutionary dynamics *within* populations (or species).

In the systematics community, taking account of population dynamics in a phylogenetic context has meant implementing some version of the *multispecies coalescent* (e.g. Liu *et al.*, 2008, 2010; Song *et al.*, 2012; Bryant *et al.*, 2012; Rannala and Yang, 2017). As discussed in Chapter 1, the coalescent models the distribution of gene trees within a population; the multispecies coalescent is its natural extension to multiple populations

or species. One of the key advantages of both the coalescent and multispecies coalescent is that, for neutral mutations, the gene tree can be decoupled from the mutation process, a feature which forms the basis of many implementations of the model.

Nevertheless, the coalescent has its limitations. It is difficult to incorporate selection, for example. Also, the running time of coalescent based methods (e.g. BEAST, *BEAST, etc.) depends critically on the number of individuals being sampled. In *BEAST a gene tree is sampled for each locus, with the number of leaves in the gene tree given by the number of individuals sampled from all populations.

Bryant *et al.* (2012) showed that explicit sampling of gene trees for each locus could be avoided in the case of unlinked binary markers. This is appropriate for the estimation of species trees from unlinked single nucleotide polymorphisms (SNPs). Their algorithm was implemented in SNAPP, and can analyse data sets with hundreds of thousands of loci and up to 200 individuals (depending on computing resources and sampling difficulties). The running time of SNAPP is $O(sn^2 \log n)$ with s species (populations) and n individuals. We note that the running time of SNAPP does not scale well as the number of individuals increase.

Rather than tinker with the SNAPP algorithm, we have taken a completely new approach with a different kind of model. Like Gutenkunst *et al.* (2010), Sirén *et al.* (2010) and Lukic and Hey (2012) we use diffusion models, though we apply them in a new way. The end result is that we can carry out SNAPP-type analyses but with essentially no limits on the number of individuals being sampled.

Diffusion models, like the coalescent, are a convenient approximation of the standard Wright-Fisher discrete models. In fact, diffusion models pre-date coalescent models by a good fifty years (Wright, 1931; Kingman, 1982a). Whereas coalescent models describe the distribution of the genealogy or gene tree for a sample of individuals, diffusion models describe the frequency of an allele in the population as a whole. There are straightforward extensions incorporating selection.

There are three challenges to overcome working with a model of allele frequencies. First, we do not actually observe the population frequencies, we just observe a sample drawn from that population. This problem is easily solved by incorporating the sampling step explicitly into our likelihood.

Second, gene frequencies are continuous traits, so it is not possible to sum over ancestral trait values as in Felsenstein's pruning algorithm (Felsenstein, 1981). For this, we follow a numerical approach developed in Hiscott *et al.* (2016), though with some new twists to improve efficiency and accuracy.

Third, diffusion models do not give explicit transition probabilities or densities: these are only available via partial differential equations (PDEs). We solve these differential equations numerically, extending a standard spectral approach from a single population to the entire species tree. We note that there is significant potential for adapting our methods to other diffusion-based models.

2.2 Modelling allele frequencies on a species tree

The diffusion model describes how allele frequencies change over time in a single population. The model extends directly to multiple populations in a species tree (Sirén *et al.*, 2010). As in Bryant *et al.* (2012) we think of the root of the species tree to be at the top of the tree and the leaves at the bottom. The model describes evolution of the allele frequencies from the top of the tree to the bottom.

The allele frequency at the root has a distribution given by the stationary density of the diffusion model (in this case, a beta distribution). The allele frequency at the bottom of a branch has a distribution given by the diffusion model with an initial value equal to the allele frequency at the top of the branch. At a speciation, the two daughter populations have the same allele frequency as the parent population.

We now formalise these ideas. Suppose that the branches in the species tree are numbered $i = 1, 2, \ldots, 2s-2$ where, for convenience, the branches adjacent to the leaves are numbered $1, 2, \ldots, s$. Let X_i^T denote the allele frequency immediately below the top of branch i. Let X_i^B denote the allele frequency immediately above the bottom of branch i. Let $X_\rho = X_\rho^B$ denote the allele frequency at the root.

At the root, ρ, the proportion of red alleles in the ancestral population is distributed according to the stationary distribution of the diffusion model. The stationary distribution of the forward diffusion (1.6) is a beta density (Ewens, 2004)

$$\pi_\rho(x|\beta_1, \beta_2) = \frac{\Gamma(2\beta_1 + 2\beta_2)}{\Gamma(2\beta_1)\Gamma(2\beta_2)} x^{2\beta_2 - 1}(1-x)^{2\beta_1 - 1}, \qquad 0 < x < 1. \tag{2.1}$$

Along any branch in the species tree, the changes in allele frequencies are modelled using the diffusion process. The allele frequency at the start of the branch gives the initial density, $f(x, 0)$, for $x \in [0, 1]$. The distribution of the allele frequency y at the end of the branch is then given by $f(y, t)$, where t is the length of the branch in units of $2N$ generations and $y \in [0, 1]$.

At a speciation, we make the assumption that there is no correlation between allele

Algorithm 2 An algorithm to simulate allele counts.

```
procedure AlleleSim(S)
    Generate X_ρ^B from density π(x|β_1, β_2) given in (2.1)
    for all nodes i except the root in a pre-order traversal do
        Set allele count X_i^T to X_j^B, where j denotes the parent node of i.
        Use diffusion to simulate X_i^B given initial value X_i^T
        if node i is a leaf then
            Generate binomial random variable r_i from Bin(2n_i, X_i^B)
        end if
    end for
end procedure
```

Simulation algorithm for allele counts under the diffusion model. Suppose S contains all parameters related to a specific species tree. Branches are numbered $i = 1, 2, \ldots, 2s-2$ where, for convenience, the branches adjacent to the leaves are numbered $1, 2, \ldots, s$. The root branch is denoted by ρ. Let X_i^T denote the allele frequency immediately below the top of branch i; X_i^B denote the allele frequency immediately above the bottom of branch i; X_ρ^B denotes the allele frequency at the root. The values $r_1, r_2, \ldots, r_s$ are the simulated red allele counts for each species. Note that in a *pre-order traversal* we visit every node in order so that the children of a node are always visited *after* the node itself.

type and speciation. Hence the two descendant species are assumed to have identical allele frequencies to the parent node.

We have, therefore, a model for allele frequencies over the whole tree. However allele frequencies at the leaves are not directly observed. Instead we have a sample of n_i individuals, giving $2n_i$ allele copies taken from each species i. If x_i is the red allele frequency for the population then the observed number r_i of red alleles in the sample for this gene has binomial distribution with parameters $2n_i$ and x_i, so that

$$P[R_i = r|x_i] = \binom{2n_i}{r} x_i^r (1-x_i)^{2n_i - r},$$

see Sirén *et al.* (2010).

Algorithm 2 simulates allele counts on a species tree under this model. Also, in Figure 2.1 we illustrate Algorithm 2 on a three taxa species tree.

We use the algorithm of Jenkins *et al.* (2017) to simulate diffusions along each branch. We give the details of the algorithm in Section 2.8.

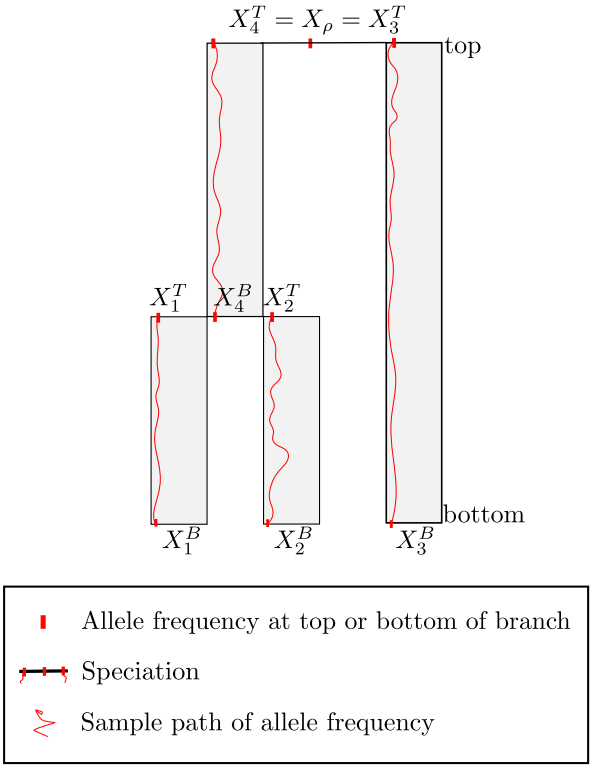


Figure 2.1: A sample path (in red) of allele frequencies along a three taxa species tree. Starting at the top of the tree we draw X_ρ from the stationary distribution (2.1) and simulate allele frequencies according to the Wright-Fisher dynamics until the leafs of a tree is reached. Thereafter we draw from binomial distributions based on allele frequencies at the leafs of the tree (X_1^B,X_2^B,X_2^B).

2.3 Analytical formula for partial likelihoods

2.3.1 Definitions

We now describe the likelihood functions analytically. For each node i and each ancestral state $x \in [0,1]$, which is our case is an allele frequency, we define two partial likelihoods, $\ell_i^B(x)$ and $\ell_i^T(x)$. The first is the probability of observing all allele counts for taxa in the subtree rooted at i, conditional on the state X_i^B at the bottom of branch i being equal to x. The second is defined in the same way, but is instead conditioned on the state X_i^T at the top of the branch. A similar approach was used in Bryant *et al.* (2012).

Next we define these partial likelihoods recursively for an individual site m.

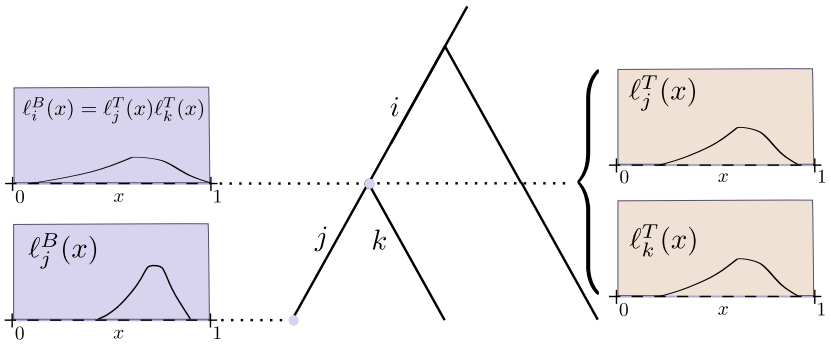


Figure 2.2: An illustration of the partial likelihoods on a species trees. We show the partial likelihood $\ell_j^B(x)$ at the bottom of a leaf branch j. Moving along the branch we also show the partial partial likelihood $\ell_j^T, \ell_k^T(x)$ at the top of the leaf branches j and k as well as the partial likelihood $\ell_i^B(x) = \ell_j^T(x)\ell_k^T(x)$ after a speciation at the bottom of the parent branch i.

2.3.2 Partial likelihoods at the leaves

Suppose that node i is a leaf and that we have sampled n_i diploid individuals from this population. Let r_i denote the number of observed gene copies carrying the red allele for site m. If x is the proportion of red alleles in the population then r_i has a binomial distribution with parameters n_i and x. Hence the partial likelihood at a leaf i is given by

$$\ell_i^B(x) = \binom{2n_i}{r_i} x^{r_i}(1-x)^{2n_i - r_i}. \tag{2.2}$$

See Figure 2.2 for depiction of partial likelihoods at the leaves.

2.3.3 Partial likelihoods at a speciation

Let j and k be the children of node i. We assume that the allele frequencies for daughter populations after a speciation are exactly those of the parent population before speciation. The partial likelihoods at the bottom of the branch above i are then the product of partial likelihoods at the tops of the branches immediately below i, so

$$\ell_i^B(x) = \ell_j^T(x)\ell_k^T(x) \tag{2.3}$$

for all $x \in [0, 1]$. See Figure 2.2 for depiction of partial likelihoods at a speciation.

2.3.4 Partial likelihoods along a branch

Let N_i be the effective population size for the branch directly above node i, and let τ_i denote the length of the branch above node i, measured in numbers of generations. Then $t_i = \frac{\tau_i}{2N_i}$ equals the length of the branch in coalescent units. Suppose that we already know the partial likelihood values at the bottom of the branch. That is, for each value of $y \in [0,1]$ the value $\ell_i^B(y)$ equals the probability of observing everything below branch i, given that the allele frequency at the bottom of branch i equals y. We would like to determine the corresponding partial likelihood at the top of the branch.

Suppose that the allele frequency at the top of the branch is x. The allele frequency changes randomly along the branch. The branch has length t_i, so we use $f_x(y, t_i)$ to denote the density of the allele frequency at the bottom of the branch, given that the allele frequency at the top is x. The function $f_x(y, t_i)$ is a density in y (possibly with point masses at 0 and 1) for each value of t_i. The partial likelihood $\ell_i^T(x)$ is the probability of observing everything below branch i conditional on the state at the top of branch i. The joint probability (density) of the allele frequency changing to y along the branch, and of observing all of the allele counts below that, is given by

$$f_x(y, t_i)\ell_i^B(y).$$

To obtain $\ell_i^T(x)$ we marginalise out over possible values for y, giving the integral

$$\ell_i^T(x) = \int_0^1 f_x(y, t_i)\ell_i^B(y)dy. \tag{2.4}$$

To get a handle on (2.4) we define a new function g, given by

$$g(x,t) = \int_0^1 f_x(y,t)\ell_i^B(y)dy, \qquad \text{for } 0 \le t \le t_i. \tag{2.5}$$

Then $g(x,t)$ is equal to $\ell_i^B(x)$ when $t = 0$:

$$g(x,0) = \ell_i^B(x), \text{ for } x \in [0,1]. \tag{2.6}$$

and $g(x,t)$ is equal to $\ell_i^T(x)$ when $t = t_i$:

$$\ell_i^T(x) = g(x,t_i), \text{ for } x \in [0,1]. \tag{2.7}$$

Furthermore, by differentiating both sides of (2.5) with respect to t, substituting the

formula (1.6) for the forward diffusion determining $f = f_x$ and applying integration by parts (see Appendix B for details) we obtain the PDE

$$\frac{\partial g(x,t)}{\partial t} = (\beta_2(1-x) - \beta_1 x)\frac{\partial}{\partial x}g(x,t) + \frac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2}g(x,t), \tag{2.8}$$

for the function g. Solving this PDE with the initial conditions (2.6) and evaluating the solution at $t = t_i$ gives the partial likelihood ℓ_i^T at the top of the branch. We will solve this PDE numerically, as detailed in the next chapter.

Note that the PDE (1.9) is in fact just the backward diffusion with a sign change. We remind ourselves that we have already observed in the previous chapter that solutions $g(x,t)$ of the backward diffusion (assuming zero-flux boundary conditions on the forward diffusion) is unique and smooth if a smooth initial condition is specified. We have also shown that boundary conditions for the forward diffusion is implicitly satisfied given $g(x,t)$ is smooth. Since we only work with initial conditions of the form (2.2) the uniqueness and smoothness of $g(x,t)$ is guaranteed and the boundary conditions satisfied.

2.3.5 Likelihood at the root of a tree

Equations (2.2), (2.3) and (2.4) can be applied to the entire tree, starting with the leaves and moving upwards towards the root. They define the root likelihood $\ell_\rho^B(x)$, which is the probability of observing all allele counts for all populations, conditioning on the allele frequency $X_\rho = X_\rho^B$ being equal to x.

To define the likelihood of the tree we now integrate over the allele frequency x at the root, multiplied by the stationary density of x given in (2.1):

$$L_m = \int_0^1 \pi(x|\beta_1,\beta_2)\ell_\rho^B(x)dx. \tag{2.9}$$

This is the probability for a single site m. The likelihood from all markers is found by multiplying the probabilities from each site together (or, in practice, summing the log-likelihoods)

$$\log L = \sum_m \log(L_m). \tag{2.10}$$

In the case of SNP data we need to compute the conditional likelihood, $L_m/(1-L_0)$, where L_0 is the probability of observing a constant site (See Bryant *et al.* (2012) and

Algorithm 3 Computes tree log-likelihoods

procedure LOG-LIKELIHOOD(X, S)
  for each locus **do**
    for each leaf node **do**
      Fit $\ell_i^B(x)$ using SNP data for the site
      Solve $\ell_i^T(x)$ using backward diffusion with $\ell_i^B(x)$ as intitial condition
    end for
    for post-order traversal over internal nodes (incl. root node) **do**
      Compute $\ell_i^B(x)$ using $\ell_j^T(x)$ and $\ell_k^T(x)$
      if at internal node **then**
        Solve $\ell_i^T(x)$ using backward diffusion with $\ell_i^B(x)$ as intitial condition
      end if
      if at the root branch **then**
        Compute $\log L_m$ using $\pi(x|\beta_1, \beta_2)$ and $\ell_\rho^B(x)$
      end if
    end for
  end for
  return $\sum \log L_m$
end procedure

Algorithm for computing the tree log-likelihoods under the diffusion model. Suppose X represent the SNP data as input for the algorithm and let S be all the tree parameters as input for the algorithm. Branches are numbered $i = 1, 2, \ldots, 2s-2$ where, for convenience, the branches adjacent to the leaves are numbered $1, 2, \ldots, s$. The root branch is denoted by ρ. Note that in a *post-order traversal* we visit every node in order so that the children of a node are always visited *before* the node itself.

Felsenstein (1992)).

We give a high level description of how to compute the log-likelihood for a species tree in Algorithm 3.

2.4 Translation of parameters

One of the more confusing aspects of working across population genetics and phylogenetics is the different ways that different communities parameterize different variables. In this section we summarise the parameters and outputs for our model, and show how they might be converted into other formulations.

There are four groups of input variables in our model. These are the variables that could conceivably be inferred from data using the likelihood function:

1. The species tree.

2. The branch lengths in the species tree. In our model, we measure the length of a branch in *number of generations*. Let τ_i denote the length of the branch i, which is the branch immediately above node i.

3. The mutation rates u and v giving the probabilities of mutating from a red to a green allele or a green to a red allele, *each generation*.

4. The effective population size N_i for each branch i (that is, the branch above node i).

Different methods for inferring species trees describe these parameters in different ways. The convention in phylogenetics is to express branch lengths in terms of *expected substitutions per site*. For neutral mutations, the rate of substitutions equals the rate of mutations. Under our model, the rate of mutations per generation is

$$\mu = P(\text{ lineage has red allele })u + P(\text{ lineage has green allele })v \tag{2.11}$$

$$= \frac{v}{u+v}u + \frac{u}{u+v}v, \tag{2.12}$$

so a branch of length τ_i generations corresponds to a branch length of $\mu\tau_i$ expected mutations per site.

Methods which ignore the branch lengths in gene trees are not able to infer both population sizes and branch lengths, and instead use a single parameter per branch, typically measured in coalescent units (e.g. Vachaspati and Warnow, 2015; Liu, Yu, and Edwards, 2010). If τ_i is the length of a branch in numbers of generations, the length of the branch in coalescent units is given by $t_i = \frac{\tau_i}{2N_i}$.

In our model, we parameterize effective population size for branch i directly as N_i. SNAPP (Bryant *et al.*, 2012), Best (Liu, 2008) and BPP (Yang, 2015) instead use $\theta_i = 4N_i\mu$ as the parameter for effective population size. Under an infinite sites model, θ_i equals the expected proportion of sequence differences between two individuals from the same population. In a finite sites model, $4N_i\mu$ is an overestimate of this expectation, due to backward mutations.

2.5 Dealing with confounded rates and rate matrices

An important issue with the diffusion model as we have described it, and indeed with many multispecies coalescent models, is that there is an identifiability problem with rates. If the mutation rates are multiplied by some constant c and at the same time branch lengths and population sizes are multiplied by $\frac{1}{c}$, the probability of the data remains the same.

One solution is to estimate the mutation rate μ beforehand and use this value, or a prior distribution around that value, to include μ as part of our model. The prior distribution for u and v is then reformulated so that (2.12) is satisfied. This strategy is used by *BEAST, where the average substitution (mutation) rate r_μ is fixed ahead of time.

An alternative strategy is to express branch lengths in terms of expected substitutions (mutations) per site and population sizes in terms of the θ parameter, as both of these are invariant to a choice of μ (Yang, 2015; Bryant *et al.*, 2012). As a consequence, the effective population sizes N_i cannot be inferred without additional information. This approach adapts well to phylogenetics, where it is customary to describe mutation rates using a normalised rate matrix, such as the Jukes-Cantor matrix (below matrix on the left) and HKY85 matrix (below matrix on the right), which read

$$\mathbf{Q} = \begin{bmatrix} - & 1/3 & 1/3 & 1/3 \\ 1/3 & - & 1/3 & 1/3 \\ 1/3 & 1/3 & - & 1/3 \\ 1/3 & 1/3 & 1/3 & - \end{bmatrix} \text{ and } \mathbf{Q} = \frac{1}{r} \begin{bmatrix} - & \pi_C & \kappa\pi_G & \pi_T \\ \pi_A & - & \pi_G & \kappa\pi_T \\ \kappa\pi_A & \pi_C & - & \pi_T \\ \pi_A & \kappa\pi_C & \pi_G & - \end{bmatrix},$$

respectively. In this matrix, the stationary probabilities for the nucleotides are $\pi_A, \pi_C, \pi_G, \pi_T$, the parameter κ controls the ratio of transitions to transversions, and the diagonal elements are chosen to make rows sum to zero. The Jukes-Cantor matrix on the left is already normalised, while in the HKY85 model r would be chosen to make the overall substitution rate equal to 1.

These matrices can be incorporated directly into Snapper, either using prior information for μ, or when using the same branch length and population parameter scheme as BPP. For any site, if X and Y are the two nucleotides present and we (arbitrarily) assign red to X and green to Y then the appropriate choices for u and v are simply

$\mathbf{Q}_{XY}$ and $\mathbf{Q}_{YX}$. For example, under the HKY85 model, if the red allele is nucleotide A and green allele is nucleotide C then the corresponding rates are $u = \mu\mathbf{Q}_{AC} = \pi_C$ and $v = \mathbf{Q}_{CA}\pi_A$. We are approximating a four-state model with a two-state model, but the error introduced should be minimal in the absence of highly divergent sequences or rapidly evolving sites.

2.6 Comparison with allele frequency spectrum methods

We note that diffusion models are already used in population genetics to approximate changes in the *allele frequency spectrum* (AFS) (Gutenkunst *et al.*, 2009; Lukic and Hey, 2012; Racimo *et al.*, 2016). For each value of x between 0 and 1, the AFS gives the proportion of sites for which the derived allele appears in the proportion x of the sample. For example, we might consider all sites where the derived allele appears in 30 of the 100 sampled genomes. If 5% of all sites fall into this category then the observed AFS value for 0.3 will be 5%.

Methods based on the AFS to infer genetic parameters for a single population are widespread in the literature (Boitard *et al.*, 2016; Lapierre *et al.*, 2017; Nielsen, 2000). In all of these analyses, it is assumed that the population dynamics have achieved stationarity when the AFS predicted by the model is used.

The *joint-AFS* is the extension of the AFS to multiple populations. Suppose that there are s species. For every vector $(x_1, x_2, \ldots, x_s)$ of numbers between 0 and 1, the joint-AFS gives the proportion of sites where the derived allele appears in a proportion x_1 of the individuals in the first population, x_2 of the individuals in the second population, and so on. While the AFS for each population by itself will be the stationary AFS, the joint-AFS for multiple populations depends on the time since separation. Fortunately the PDE determining the predicted joint-AFS is a straight-forward extension of the PDE for a single population (Gutenkunst *et al.*, 2009; Lukic and Hey, 2012; Racimo *et al.*, 2016).

There are two main advantages of an approach based on the joint-AFS when compared to the method we describe here. First, the PDE for the joint-AFS only needs to be solved once for all sites, whereas in our approach we end up solving the PDEs once for each (distinct) site. Second, migration between populations can be incorporated quite simply into AFS calculation, whereas it would break down the dynamic programming strategy that we use.

The main disadvantage of the joint-AFS strategy is that the PDE has as many dimensions as the number of species, and suffers from the *curse of dimensionality* (Gutenkunst *et al.*, 2009), leading to an exponential growth in the size of grid used by standard numerical methods. This factor severely limits the number of species which can be considered concurrently, whereas in our approach the algorithm scales linearly with the number of species.

2.7 Snapper

"No more buddy-buddy, no more messing around." (Gutteridge et al.*, 1988)*

2.7.1 Overview of numerical implementation

So far, we have only discussed how the model and likelihood are defined analytically, not how they are computed. Making the algorithm efficient for large-scale analysis required development of a large suite of numerical algorithms, many of which are novel. Details of these methods are given in the next chapter; here we outline the major features.

Since we cannot store the partial likelihood functions ℓ_i^B and ℓ_i^T in their entirety, we instead approximate them using a finite number, K, of basis functions, in our case the shifted Chebyshev functions (details in the next chapter). These are simple functions, defined in advance, and we approximate the partial likelihood functions using a weighted sum of the basis functions, the weights being determined as we move up the tree. As the number K of basis functions increases, the accuracy increases, though so does the computation time. The approach using basis functions differs from that of Hiscott *et al.* (2016), where values of the functions at a mesh were used instead. The new approach is more flexible and has greatly improved accuracy, as we demonstrate below. In fact it is possible to bound the error introduced in approximation and show that this error decreases extremely rapidly as the number K of basis functions increases.

To compute the approximations of the likelihood and partial likelihood functions we substitute the approximation formulas into equations (2.2), (2.3) and (2.4), detailed in the next chapter. In practice, a lot more algorithmic development was required to obtain sufficient accuracy in speed. This included:

- Special recurrence formulas for stable evaluation of partial likelihoods at the tips (2.2).

- An efficient algorithm for computing the product of two approximations, based on fast Fourier transforms.
- Efficient algorithms which take advantage of structure in the PDE in equation to find rapid and accurate numerical solutions.
- Dynamic caching algorithms to share partial likelihood calculations between sites.

See the next chapter for details.

With all of the efficiency gains, the SNAPPER algorithm takes only $\mathcal{O}(sK\log(K))$ time per site where s is the number of species, K the number of Chebyshev basis functions. We assume that the data is pre-processed so that for each site we have the frequencies of individuals with each type of allele. This pre-processing takes $O(n+s)$ time, where n is the number of individuals, and only needs to be carried out once per data set. In comparison, SNAPP takes $\mathcal{O}(sn^2\log n)$ time per site with same amount of pre-processing. The algorithms scale differently but more importantly the running time of SNAPPER depends on the number of basis functions K, not on the number of individuals.

The likelihood algorithm forms the core of a Bayesian inference software package, SNAPPER. The software is open-source and available for download at `https://github.com/rbouckaert/snapper`. Like SNAPP, it takes biallelic data at multiple loci for multiple individuals in a set of species and returns samples from the joint posterior distribution of (i) species phylogenies, (ii) species divergence times and (iii) effective population sizes. As in SNAPP we implemented multithreading to take advantage of parallel computation on multiple core machines or graphics processing units. The range of prior distributions, and flexible prior specification, remains essentially unchanged (we discuss some unusual priors in the next section).

The MCMC proposal function implemented is almost the same subset of BEAST2 (Bouckaert *et al.*, 2019) move proposals as that implemented for SNAPP. We add one new proposal that selects a subtree of the species tree and scales all population sizes within that subtree simultaneously, a refinement of an existing proposal.

The user can specify the number K of basis functions. Some recommendations are made in the SNAPPER manual regarding the number of basis functions to use given the size of a dataset. Also contained in the software package are simulators and python scripts to integrate SNAPPER with the iPyRad (Eaton and Overcast, 2016) data pipeline to assist with streamlining data analysis.

2.7.2 Special priors

Yule prior on species trees

The Yule branching process (Yule, 1925) is a pure birth process where each branch is fitted with the bifurcation rate λ. The exponentially distributed waiting times between splits define the prior distributions for the branch lengths of the species tree.

Suppose that $t_2, \ldots, t_s$ are the times between between splits for a species tree with s number of taxa and $2s-2$ number of branches. We start at the two child branches of the root with two independent birth processes with constant rate λ. This gives a waiting time distribution until first split (speciation) as an exponential distribution with rate 2λ. Similarly, waiting time t_i until the i-th split for $i = 2, \ldots, s-1$ is distributed as an exponential distribution with rate $i\lambda$. The waiting time interval t_s is defined in terms of a Poisson distribution with rate s of observing no events. The prior distribution on the species tree is then defined in terms of tree length (sum of all branches) given number of species s,

$$p(t_2, \ldots, t_s|s) = (s-1)!\lambda^{s-2} \exp -\lambda \sum_{i=2}^{s} it_i,$$

see Nee (2001).

Cox-Ingersoll-Ross prior on population size

The Cox-Ingersoll-Ross (CIR) process is widely used in finance to model interest rate fluctuations (Cox *et al.*, 2005). Lepage *et al.* (2006) use the process to model fluctuations of evolutionary rate along a tree. The CIR process is one of the few mathematically tractable random processes which are continuous, positive, and have a stationary (long term) distribution. At any point in time, the process has a gamma distribution. If we condition on current value then the distribution at some time t in the future is non-central Chi-squared. The equations are quite messy - see Lepage *et al.* (2006) for further details.

We use the process to model the prior distribution of population parameters θ over the tree. The value of θ in the root population is assumed to have a gamma distribution (as in BPP (Yang, 2015)). We then assume that the θ values evolve down each branch following the CIR process. However rather than integrate out the variation along each branch we just sample the values at the nodes themselves.

We therefore have a prior model with three parameters. Two are the standard

parameters α, β for the gamma distribution, and the third, κ controls the rate at which correlation between θ values disappears over time. The correlation between the values at either end of a branch of length t is given by $e^{-\kappa t}$. If κ is zero then all branches will have the same θ value; if κ is infinite then all branches will all have independent θ values.

2.8 Simulation experiments

2.8.1 Simulating allele frequencies under the Wright-Fisher model

Here we examine the problem of simulating allele frequencies under the diffusion model. Standard methods for simulating allele frequencies under the diffusion model fail due to asymptotics at the boundary. Reasonably accurate methods for simulating allele frequencies from the forward diffusions include truncating a spectral expansion of the transition function (Song and Steinrücken, 2012; Steinrücken *et al.*, 2014) or numerical solutions of the Kolmogorov equations (Williamson *et al.*, 2005; Bollback *et al.*, 2008; Gutenkunst *et al.*, 2009). Care needs to be taken when implementing these methods since approximation errors close to the boundary can become large, a problem that Jenkins *et al.* (2017) explicitly avoid.

In Algorithm 4 we give the steps of the underlying methods developed by Jenkins *et al.* (2017) to simulate allele frequencies on a branch exactly. Refer to Jenkins *et al.* (2017) for more detail on how it works. Briefly, the method exploits an eigenfunction expansion to represent the density function (Griffiths and Spano‘, 2010). The expansion admits a probabilistic interpretation and therefore lends itself to simulation techniques, but it is not straight-forward to implement because the distribution involved is only known as an infinite series. Despite this hurdle, (Jenkins *et al.*, 2017) show that it is in fact possible to simulate allele frequencies without error. Note that the algorithm given here is the implementation for step 5 in Algorithm 2.

We give the formulas for functions in Algorithm 4:

$$a_{km}^{\beta_1+\beta_2} = \frac{(\beta_1+\beta_2+2k-1)\Gamma(\beta_1+\beta_2+m+k-1)}{\Gamma(\beta_1+\beta_2+m)m!(k-m)!},$$

$$b_k^{(t,\theta)}(m) = a_{km}^{\beta_1+\beta_2} e^{-k(k+\beta_1+\beta_2-1)t/2},$$

Algorithm 4 Exact simulator for forward diffusion along a branch

procedure $C(t, \beta_1, \beta_2, m)$
    Set $i \leftarrow 0$
    while $b_{i+m+1}^{(t,\beta_1+\beta_2)}(m) < b_{i+m}^{(t,\beta_1+\beta_2)}(m)$ **do**
        $i \leftarrow i + 1$
    end while
     return i
end procedure
procedure $A(t, \beta_1, \beta_2)$
    Set $m \leftarrow 0$, $k_m \leftarrow 0$, flag $\leftarrow false$
    $U \sim$ Draw from Uniform[0,1] distribution
    while flag is $false$ **do**
        Set $k_m \rightarrow C(t, \beta_1, \beta_2, m)/2$
        while $S_{k_m}^{-}(m) < U < S_{k_m}^{+}(m)$ **do**
            Set $k_m \leftarrow k_m + 1$
        end while
        if $S_{k_m}^{+}(m) < U$ **then**
            Set $m \leftarrow m + 1$
        else $S_{k_m}^{-}(m) > U$
        end if
    end while
     return m
end procedure
procedure SIMULATE(t,β_1, β_2,x_0)
    $m \sim$Simulate $A(t, \beta_1, \beta_2)$
    $l \sim$ Draw from Binomial(m, x) distribution
    $y \sim$ Draw from Beta$((\beta_1 + \beta_2)/2 + l, (\beta_1 + \beta_2)/2 + m - l)$ distribution
    return x_t
end procedure

We simulate allele frequencies along a branch using methods in (Jenkins *et al.*, 2017). The procedure SIMULATE takes branch length (t), forward mutation rate (β_1),backward mutation rate (β_2) and initial allele frequency at top of the branch (x_0) as input and return allele frequency at the bottom of a branch. We list the formulas needed at the end of the section.

$$S^{+}_{k_m}(M) = \sum_{m=0}^{M} \sum_{i=0}^{2k_m+1} (-1)^i b^{(t,\beta_1+\beta_2)}_{m+i}(m),$$

$$S^{-}_{k_m} = \sum_{m=0}^{M} \sum_{i=0}^{2k_m} (-1)^i b^{(t,\beta_1+\beta_2)}_{m+i}(m).$$

2.8.2 Protocol

We conducted a number of simulation experiments to assess the performance of the algorithms and their equivalence with approaches based on the coalescent.

The first experiment addresses the extent of numerical error. The accuracy and running time of SNAPPER both depend heavily on the number of Chebyshev basis functions (K) used in approximations. An important question is how large the number of basis functions needs to be and the rate at which the numerical approximation converges to the true values.

We selected ten 4-taxa species trees with a caterpillar topology and ten 4-taxa species tree with a balanced topology. Half of each group of trees were scaled to have very short branches (average of 0.005 coalescent units). The other half of each group was fixed with very long branches (average of 0.5 coalescent units). We followed the same procedure for ten 16-taxa species trees with a caterpillar topology and ten 16-taxa species tree with a balanced topology. Population size (θ) parameters for each tree were generated from a Gamma distribution with mean 0.1 and variance 0.0001. For each tree we simulated data under the diffusion model for a single site drawing the total number of individuals at each tip from a uniform distribution between 5 and 30. We then computed the log-likelihood for different numbers of basis function, $K = 5, \ldots, 50$. Note that for this experiment we do not limit number of Chebyshev basis functions (K) to values of the form $2^m + 1$. Since there is no analytical expression for the likelihood we assess convergence by comparing values calculated to those with a large ($K = 200$) number of Chebyshev basis functions.

The second experiment examines the differences between the coalescent model and the diffusion model. For this experiment we simulated data according to the multi-species coalescent but analysed them using SNAPPER and a diffusion model. Theory suggests that the coalescent and diffusion models are approximations of each other (and of the underlying Wright-Fisher model).

We generated 300 species trees using the Yule distribution with speciation rate of 10. For one third of those species trees we generated population size (θ) parameters from a Gamma distribution with mean 0.01 and variance 0.0001; for another third we

generated population size (θ) parameters from a Gamma distribution with mean 0.01 and variance 0.00001; for another third we generated population size (θ) parameters from a Gamma distribution with mean 0.01 and variance 0.000001. We then simulated 1000 SNPs for each tree (with 32 individuals) under the coalescent model using the program simSNAPP (freely available at `https://www.beast2.org/snapp/`). For each simulated SNP dataset we ran two MCMC chains for 100,000 iterations each using SNAPPER. One of the chains were specified with correct priors, i.e distributions used to simulate the SNPs. For the other chain we specified incorrect priors. The incorrect prior for the tree was a Yule distribution with speciation rate of 1. The incorrect prior specified on the population size (θ) parameters was a Gamma distribution with mean 0.1 and variance 0.0001.

In the third experiment we looked at how computational time for SNAPP and SNAPPER scaled in terms of number of individuals. We simulated data according to the multi-species coalescent and compared computational time of the log-likelihood between SNAPP and SNAPPER. The SNAPPER algorithm takes $\mathcal{O}(sK\log(K))$ time per site with $\mathcal{O}(ns)$ pre-processing, where s is the number of species, K the number of Chebyshev basis functions and n is the number of individuals at a site. SNAPP takes $\mathcal{O}(sn^2\log n)$ time with same amount of pre-processing. The algorithms scale differently but more importantly the running time of SNAPPER does not increase with the number of individuals. Rather it depends on the number of Chebyshev basis functions (K).

We generated six 4-taxa species trees from a Yule distribution with speciation rate 10. We started with 10 individuals for the first tree and incrementally increased the number by 5 up to a total number of 35 individuals. For each tree we simulated a 1000 SNPs under the coalescent model. We follow the same procedure for six 8-taxa species trees. The number of Chebyshev basis functions (K) for SNAPPER was fixed at 33. In both cases the log-likelihood computation was done without caching.

All simulations and inference were run on a laptop with an Intel i7-8565U CPU.

2.8.3 Results

We summarize the convergence results of the first experiment in Figure 2.3 (4 species trees) and in Figure 2.4 (16 species trees). We see that for all trees the error of the log-likelihood decreases exponentially with number of Chebyshev basis functions (K), that is, it decreases like α^K for some $\alpha < 1$. Error is smaller for long branch lengths since approximate solutions are typically lower degree polynomials. However when the branch lengths are very short the error is greater. There are good reasons for this.

Firstly, when the partial likelihood function is spiked, the approximate solutions are high degree polynomials requiring more basis functions. Secondly, population sizes for short branches are intrinsically more difficulty to estimate, no matter which model or method is used. The only information we have about population sizes comes from the distribution of coalescent events, and on short branches there are simply insufficient coalescent events to make sound inference. Later, we address this by adopting a prior distribution on population sizes which introduces correlation between neighbouring branches. Apart from branch length distribution, tree shape does not seem to have a noticeable effect on the error convergence.

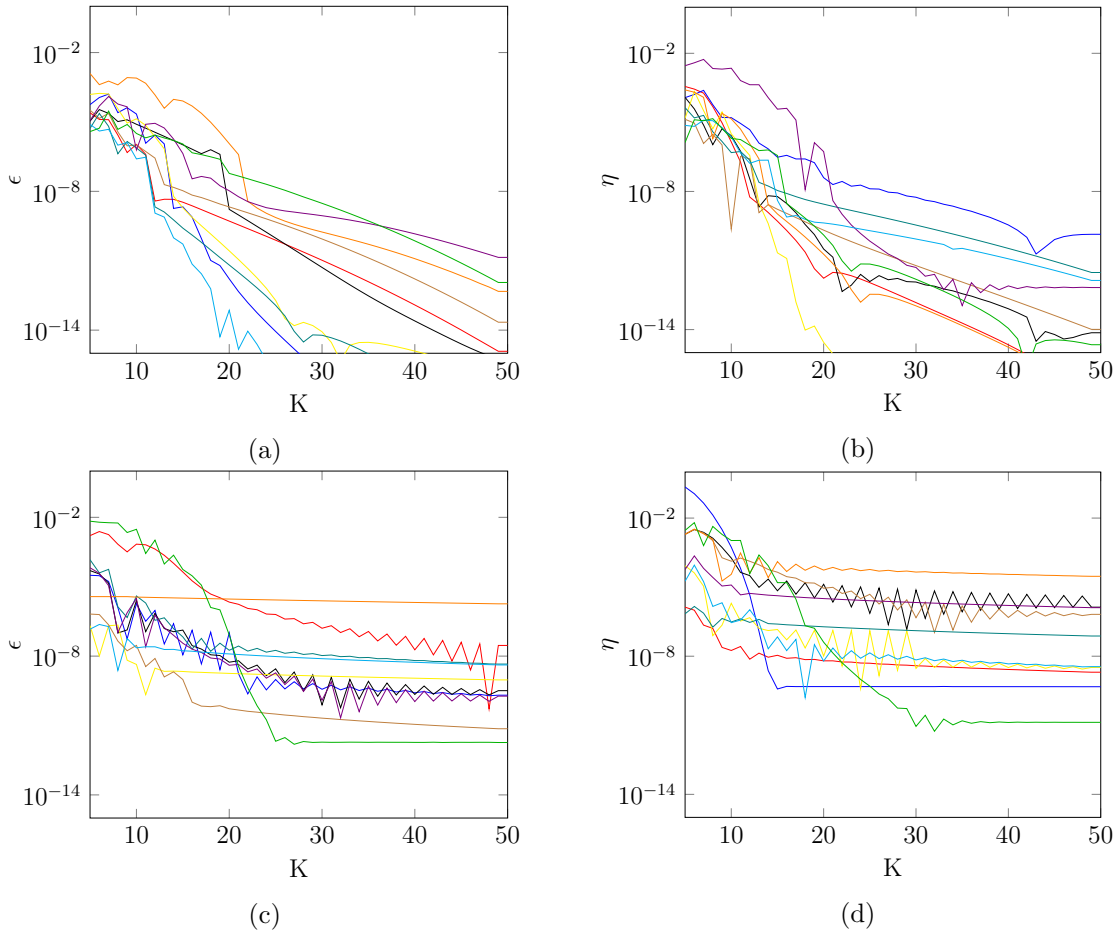


Figure 2.3: Log scale plot of relative error for basis functions $K = 5, 6, \ldots, 50$ of 4-taxa trees: (a) balanced short; (b) caterpillar short; (c) balanced tall; (d) caterpillar tall. The sub-linear decrease in log-error corresponds to an exponential decrease in error, until the limit of machine precision is reached and no further improvements in error are possible.

For the second experiment we report the frequencies for which the 95% highest posterior density of the MCMC chains generated under the diffusion model were able to recover the known parameters, see Table 2.1. We note that in both cases of correct

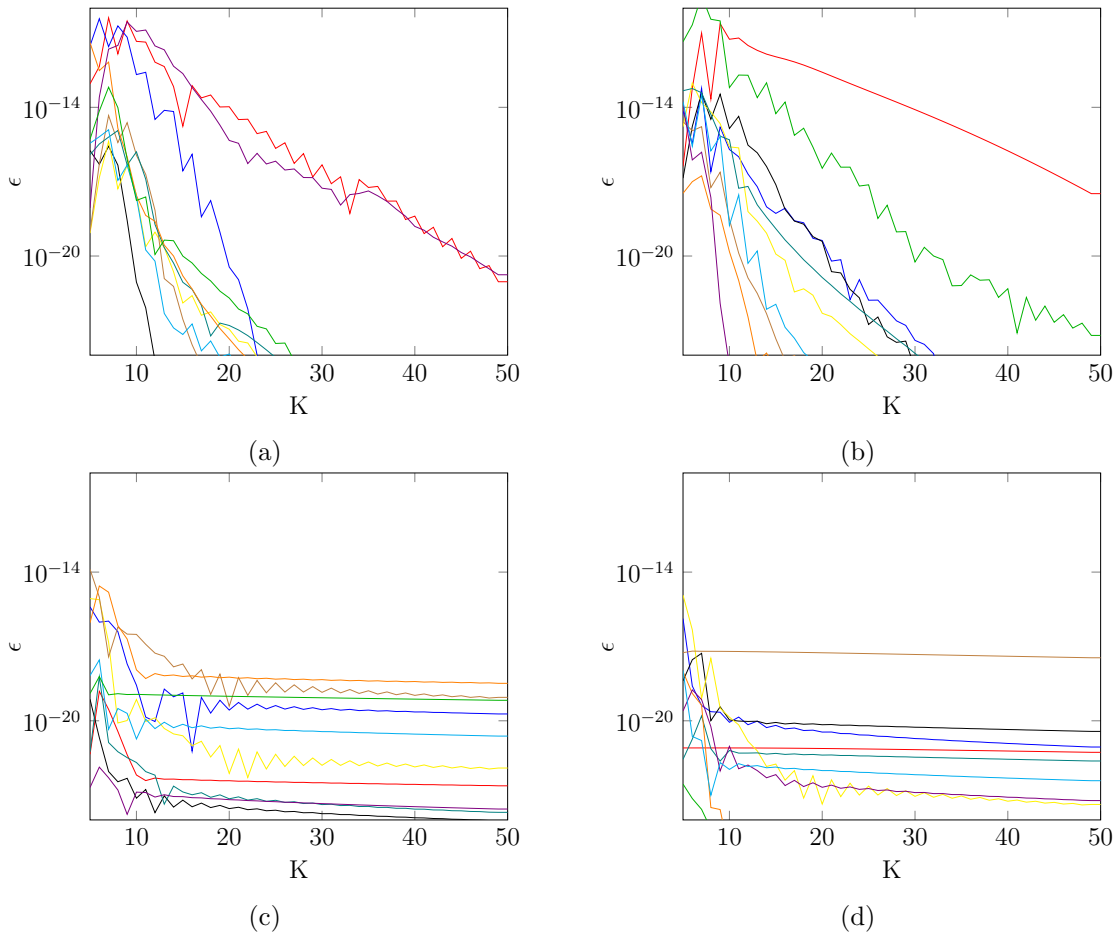


Figure 2.4: Log scale plot of relative error for basis functions $K = 5, 6, \ldots, 50$ of 16-taxa trees: (a) balanced short; (b) caterpillar short; (c) balanced tall; (d) caterpillar tall. The sub-linear decrease in log-error corresponds to an exponential decrease in error, until the limit of machine precision is reached and no further improvements in error are possible.

and incorrect priors tree height, tree length and topology is easily recovered. However we see that the SNP data generated from Gamma distributions with high variance made it harder to recover population size (θ) parameters. We can also see a slight influence of the priors specified when looking at populations size (θ) parameter recovery rates. Most notably when incorrect priors are used recovery rates for population sizes on branches near the root decrease. It is not surprising that recovery rates of populations sizes (θ) parameters are low close to the root. This is not due to the different models that were used for simulating data (coalescent model) and inferring parameters (diffusion model). Rather it is because posterior variance on population size rapidly increase as we move along branches up the tree. Equivalently we can say that there is very little information in the SNP data for population sizes close to the root. In contrast we see that SNP data contains a lot of information about the height, tree length and topology

Sim	Priors	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_ρ	height	length	top.
Γ(2,200)	C	0.91	0.91	0.94	0.93	0.94	0.97	0.95	1.00	1.00	1.00
Γ(2,200)	I	0.84	0.88	0.87	0.84	0.75	0.71	0.06	0.99	0.99	1.00
Γ(4,400)	C	0.93	0.97	0.92	0.97	0.9	0.98	0.95	1.00	1.00	1.00
Γ(4,400)	I	0.91	0.95	0.93	0.94	0.78	0.68	0.32	0.99	0.99	1.00
Γ(80,8000)	C	0.98	0.97	0.99	0.99	0.97	0.98	0.97	1.00	1.00	1.00
Γ(80,8000)	I	0.94	0.95	0.96	0.93	0.84	0.83	0.51	1.00	1.00	1.00

Table 2.1: We summarize the frequency that parameters fall within the 95% highest posterior density for simulations from the second experiment. In the "Sim" column we indicate the Gamma distributions used for simulating SNP data. In the "Prior" column "C" indicates correct priors used simulating MCMCs and "I" indicate incorrect priors used for MCMC simulation. Height is the height of the tree; length is the total sum of the branch lengths; top. is the topology of the tree.

since recovery rates are unaffected by priors. The MCMC chains took on average ∼200 seconds to run.

We give computational times of SNAPP and SNAPPER from the third experiment in Figure 2.5. We see that SNAPP scales roughly $\mathcal{O}(sn^2 \log n)$ as the number of individuals (n) increase. This was the biggest drawback of SNAPP. We see that SNAPPER no longer suffers from this issue since computation time is not affected at all by the number of individuals (n). As we have shown in the previous experiment convergence took slightly longer when the number of individuals increased. Therefore we would recommend that the number of basis functions is set to the maximum (i.e $K = 33$) when large number of individuals are used in an analysis where very short trees are expected.

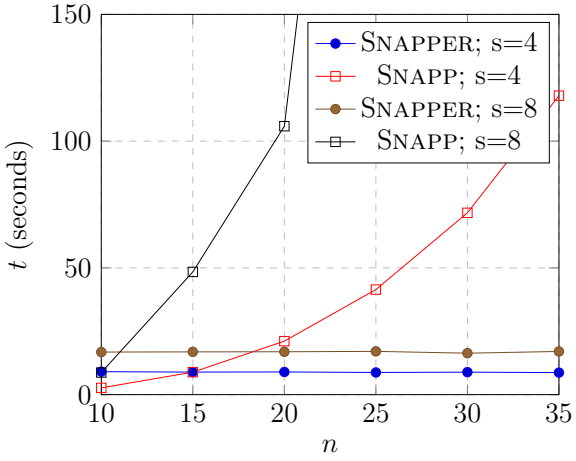


Figure 2.5: Average times (in seconds) to compute the likelihood of a 1000 sites on 4- and 8-taxa trees, for SNAPP and SNAPPER with $s = 4, 8$ (number of taxa) and $n = 10, 15, \ldots, 35$ (number of individuals).

2.9 Analysis of wild and cultivated soybeans

2.9.1 Description of soybean dataset

Our simulation experiments confirm that the multispecies coalescent model underlying SNAPP and the diffusion model behind SNAPPER are approximations of one another (Griffiths and Spano‘, 2010), with differences decreasing as effective population size increases. To demonstrate this in practice we analyzed the soybean dataset in Chifman and Kubatko (2014). The data consists of 1,027,026 SNPs from a total of 20 individuals in 10 populations.

In Chifman and Kubatko (2014) it was reported that SNAPP was not sampling from the posterior distribution efficiently enough. It quickly became apparent that the main cause of difficulty was the inference of population sizes on branches with close to zero length. We selected the CIR prior in SNAPP and in SNAPPER, which implements a model with correlated population sizes on nearby branches. We also implemented a proposal which scales all population sizes within a randomly selected subtree.

Priors for the analysis were specified as follows. We assumed a prior Yule distribution on the species tree with rate 3.85794 and average tree height of 0.5 coalescent unit. We used a correlated CIR prior for population sizes with hyperparameters $\alpha = 2$, $\beta = 200$ and $\kappa = 10$.

We then ran 10 MCMC chains for 1,000,000 iterations each for both SNAPP and SNAPPER.

2.9.2 Results

It took SNAPP approximately ~ 161 hours running on 32 threads to complete a chain simulation. It took SNAPPER approximately ~ 152 hours using 32 threads to run. Note that due to small number of individuals per population SNAPPER has little advantage over SNAPP in terms of runtime for this particular dataset. Tracer (Rambaut *et al.*, 2018) was used to assess convergence by looking at trace plots and effective sample size for each parameter. We give the summary statistics in the appendix (A.3).

Figure 2.6 (SNAPPER analysis) and Figure 2.7 (SNAPP analysis) summarize the posterior distribution of the species tree using Densitree (Bouckaert, 2010). We print the posterior mean of effective population size above each branch. In both cases we see that there is only one topology in the 95% highest posterior density. Posterior distributions of branch lengths are mostly indistinguishable. There are some small differences

in posterior distributions of populations sizes. However in all cases population sizes follow the same apparent distributions.

2.10 Analysis of freshwater turtle; Emydura macquarii

2.10.1 Description of freshwater turtle dataset

To illustrate the application of SNAPPER to large datasets we reanalyse unlinked SNP data of Georges *et al.* (2018) from a group of freshwater turtles known collectively as *Emydura*. The range of Emydura extends almost the full length of the Australian continent from north to south. The group is currently recognized as a complex of closely related and morphologically distinct allopatric forms, variously regarded as species, subspecies or distinct morphological lineages (Georges *et al.*, 2018).

The dataset consists of samples from 399 individuals divided into 41 populations and sampled from 57 distinct water bodies. The sampling covers the coastal drainages of eastern Australia, from the Hunter River in the south (New South Wales) to the Normanby River (Queensland) in the north; the rivers of the Murray-Darling Basin (MDB), including the Paroo drainage, and the Lake Eyre Basin (LEB), and the intervening Bulloo Basin (Georges *et al.*, 2018). The analyzed dataset contains 5,186 unlinked SNPs after sites with missing data was removed.

Priors for the analysis were specified as follow: We assumed a prior Yule distribution on the species tree with rate 3.30293 and average tree height of 1 coalescent units. We used a correlated CIR prior for population sizes with hyperparameters $\alpha = 2$, $\beta = 200$ and $r = 10$.

We ran the SNAPPER sampler for 2,000,000 iterations with convergence assessed in Tracer (Rambaut *et al.*, 2018).

2.10.2 Results

It took a total of ~ 500 hours for the sampler to run on a computer with an Intel i3-7100 CPU. We provide a complete list of summary statistics in the appendix (A.3). The shape of the densitree in Figure 2.8 agrees with the genetic distances and SVDquartets trees computed in Georges *et al.* (2018). We extend the anaylsis in Georges *et al.* (2018) by quantifying uncertainty around branch lengths as well as introduce popula-

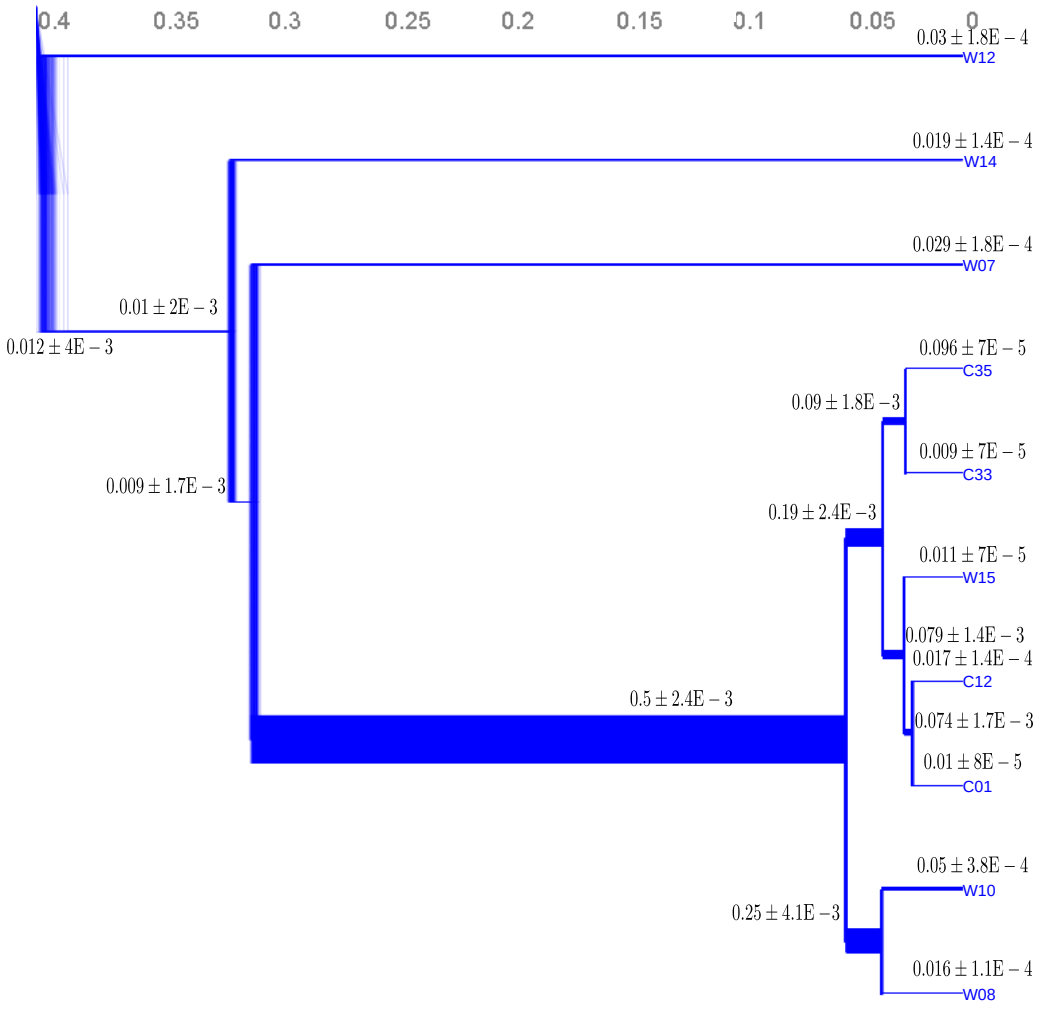


Figure 2.6: A SNAPPER inference of soybean species trees displayed using Densitree (Bouckaert, 2010). Branch thickness is related to relative population sizes, tree height is reported in expected number of mutations and population sizes is printed below each related branch, as $\mu \pm \sigma$

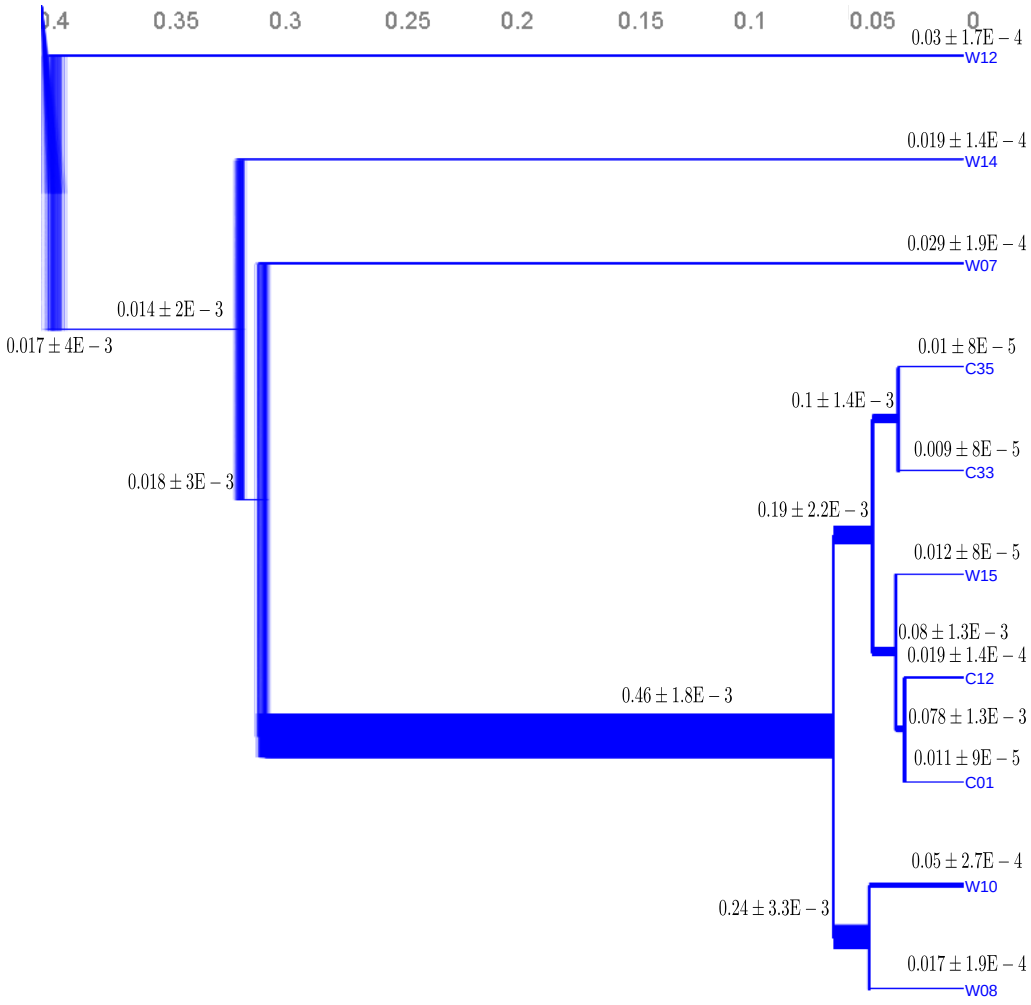


Figure 2.7: A SNAPP inference of soybean species trees displayed using Densitree (Bouckaert, 2010). Branch thickness is related to relative population sizes, tree height is reported in expected number of mutations and population sizes is printed below each related branch, as $\mu \pm \sigma$

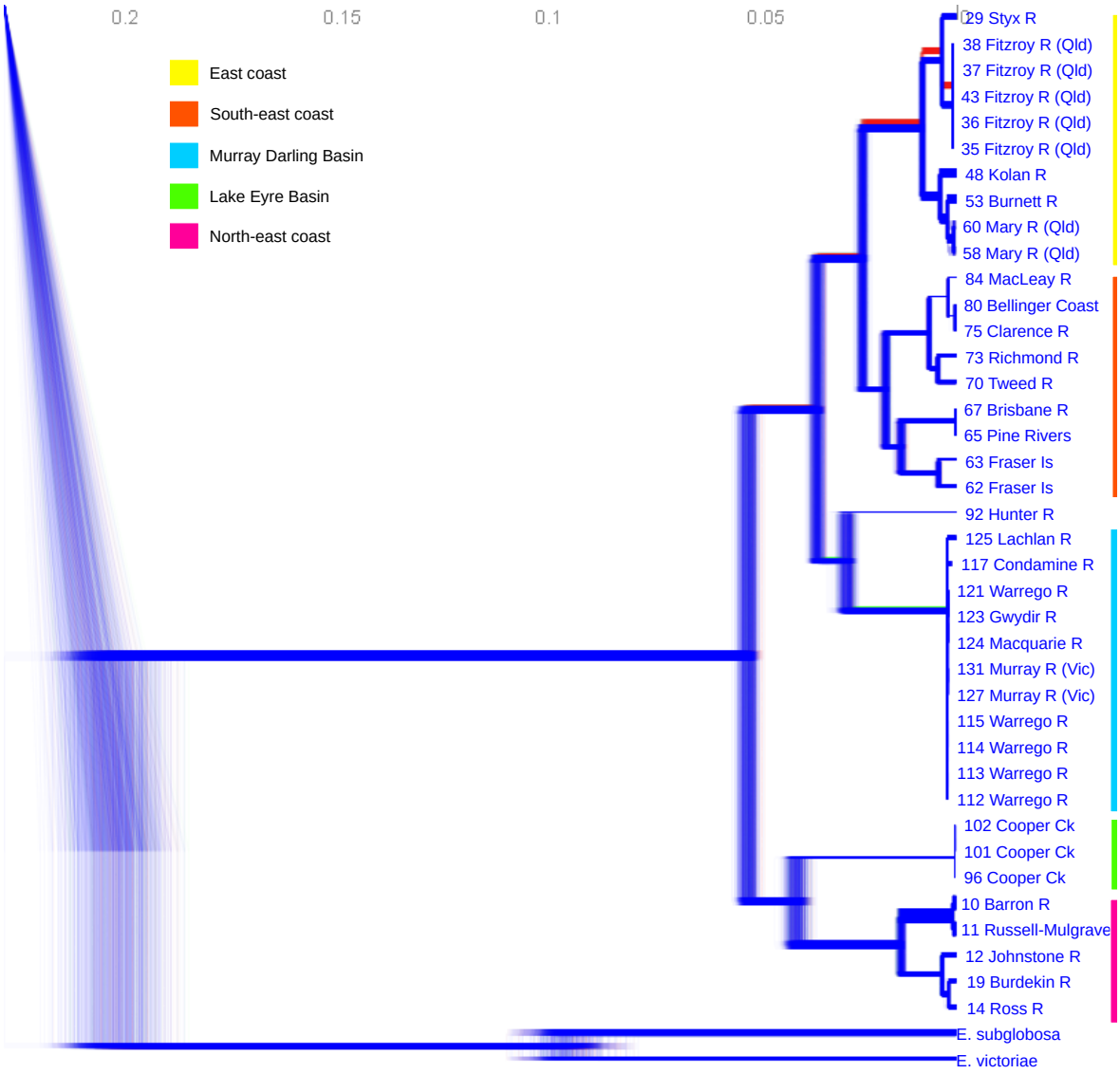


Figure 2.8: 41 population densitree of freshwater turtle *E. macquarii*. On the x-axis, variation in the tree represents uncertainty of branch lengths. Thickness of the branches represent posterior mean of population sizes. Timescale grid at the top is given in expected number of mutations per lineage per site.

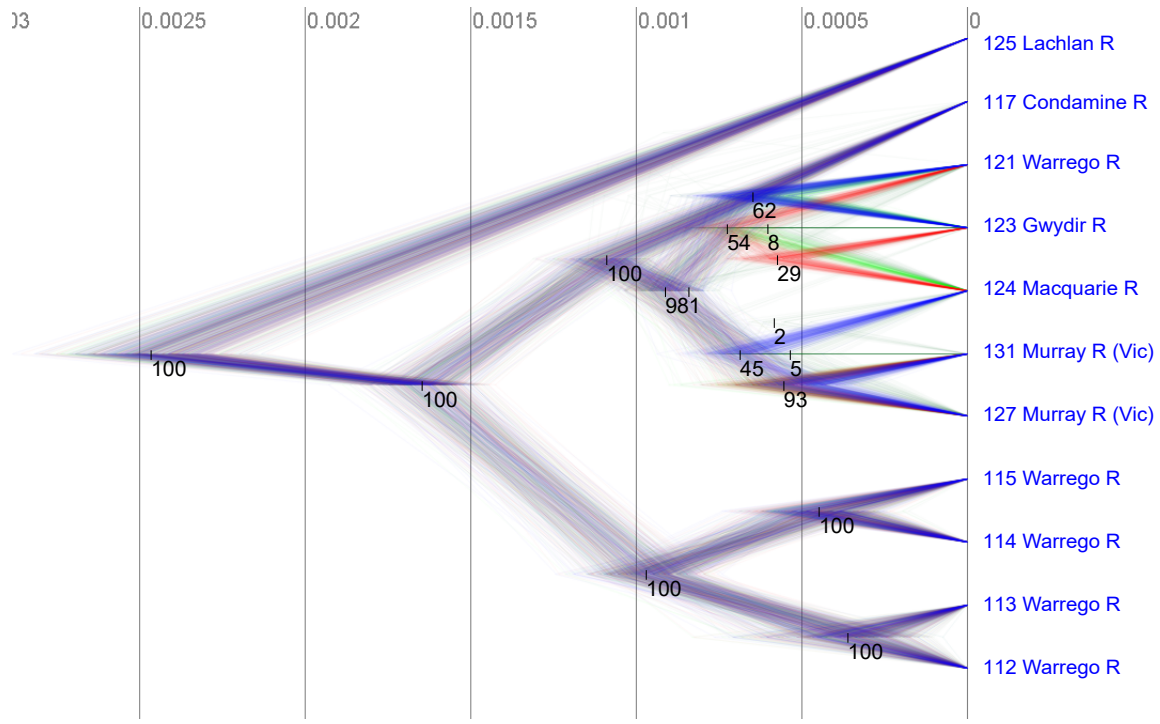


Figure 2.9: 11 population densitree of freshwater turtle *E. macquarii* restricted to MDB clade to provide better resolution of branch lenghts and topology. We show the fraction of trees in the tree set that contain a clade as text on the graph. We also display the mean of a node as a marker on the tree. Timescale grid is given in expected number of mutations per lineage per site.

tion size (θ) parameters. In Figure 2.8 we see some uncertainty surrounding topology in the Fitzroy clade (samples 38 - 43). The Cooper Creek clade (samples 96 - 102) is sister to the North-east Coast clade (samples 10 - 19) in all three analyses, as is the sister relationship between the Hunter R (sample 92) and the Murray-Darling Basin (samples 96 - 115) populations. The East Coast (samples 29 - 60) and South-east Coast (samples 11 - 20) clade is sister to the Hunter-MDB clade (samples 20 - 31) in the SNAPPER tree, in agreement with the Fitch-Margoliash distance tree. Sensibly, the Kolan-Burnett-Mary (samples 48 - 58) clade is sister to the Fitzroy clade in the SNAPPER tree. The SNAPPER analysis supports relationships within the NE Coast clade that reflect drainage proximity better than the SVDquartets analysis.

Also, we note that *E. subglobosa* and *E. victoriae* populations are considered outgroups. Therefore we expect to see the divergence time from the rest of the populations to be much earlier. The early divergence time for the two outgroups leads to poor resolution for the MDB clade (samples 112 - 131). Thus we present the MDB clade in Figure 2.9. The figure also depicts the extent of uncertainty surrounding the tree due to short branch lengths. As we discuss above, population size estimates here will be

mostly dependent on the prior due to little information available on such short branch lengths.

2.11 Discussion

In this chapter we present SNAPPER, a computationally more efficient method to supersede SNAPP. Like SNAPP the method takes biallelic markers sampled from individuals in multiple populations and computes the likelihood of the species tree topology together with branch lengths and population sizes. We achieved this computational efficiency by computing the likelihood using diffusion models rather than the multispecies coalescent. The Wright-Fisher diffusion and Coalescent are dual processes and it is therefore not surprising that the same inference can be made under these distinct but related model frameworks.

We utilize observed allele frequencies in the sample as initial conditions for the backwards diffusion. The advantages of using the backward diffusion are two-fold. Firstly the numerical solutions are stable and bounded. Secondly, it is clear how to define the boundary conditions.

The SNAPPER sampler is based on the SNAPP sampler and uses the same move proposals, which are standard in the Beast2 software package. However to improve sampling efficiency for large trees we implemented an additional move to scale population sizes on subtrees.

We have reported some of the analyses performed in order to validate and more importantly convince the reader of the ability of SNAPPER to infer population genetic parameters.

Finally, we note that the general framework for defining likelihoods in terms of backward diffusions can be extended to other diffusion models. These models can include additional parameters such as selection, migration and linkage disequilibrium.

Chapter 3

Mathematical machinery of Snapper

3.1 Introduction

In the previous chapter, we showed how the recursions for the partial likelihoods can be derived analytically. We have also shown the capability of diffusion models and how it scales efficiently for SNP data sets with large number of taxa. However we did not show how to actually compute those diffusion model likelihoods. Indeed, computing the likelihoods amounts to an extremely high dimensional integration problem. Our approach is to use numerical techniques combined with dynamic programming. While the likelihoods we compute are approximate, we can still bound the error.

The approximation method we use for the backward diffusion is part of a larger class of PDE approximation methods known as *spectral methods*. See Trefethen (2000) for a general introduction to spectral methods. Roughly speaking the idea behind spectral methods is that PDEs can be reduced to ordinary differential equations (ODEs) by expressing the approximate solution as a combination of smooth functions called *basis functions*.

In this chapter we take a look under the hood of Snapper. We spend time to discuss the detail behind the efficient likelihood computations. We begin by describing a class of basis functions known as *shifted Chebyshev polynomials of the first kind*. The properties of shifted Chebyshev polynomials are key to the numerical algorithm. We derive approximate ODE solutions of the backward diffusion in terms of shifted Chebyshev polynomials. Our derivation is less general than typical Galerkin approximations (Douglas and Dupont, 1970). Furthermore we derive explicit formulas for the ODE

systems. We outline some neat mathematics for approximating matrix exponentials. We also use properties of shifted Chebyshev polynomials to set up efficient integration methods at the root of a species tree, which was challenging due to singularities of the stationary distribution (2.1). Lastly, we package all of the accumulated numerical results and computational tricks into an algorithm to solve diffusions on species trees which runs linear in the number of species, linear in the number of sites and linearithmic ($K \log K$) in the number of basis functions. This algorithm underlies the Bayesian software package SNAPPER.

3.2 Shifted Chebyshev polynomials

3.2.1 Definitions and properties

Hiscott *et al.* (2016) describe a general strategy for computing likelihoods numerically whereby partial likelihoods are evaluated on a mesh of values at each node and accurate quadrature methods used to carry out the actual computation. We will extend the same strategy by using shifted Chebyshev polynomials as a set of basis functions to express approximate partial likelihoods instead of a mesh of values. The basis functions are chosen to help solve the backward diffusion (1.9) efficiently and accurately.

We denote shifted Chebyshev polynomials as

$$T_0^*(x), T_1^*(x), T_2^*(x), \ldots.$$

Shifted Chebyshev polynomials are defined on $[0, 1]$ and have particularly useful properties some of which we discuss below (Fox and Parker, 1968). They also have a lot of equivalent definitions, but the simplest uses the Cosine function

$$T_k^*(x) = \cos k(2x - 1).$$

They also satisfy the recursion

$$T_0^*(x) = 1, \tag{3.1}$$

$$T_1^*(x) = 2x - 1, \tag{3.2}$$

$$T_k^*(x) = 2(2x - 1)T_{k-1}^*(x) - T_{k-2}^*(x), \qquad k = 2, 3, \ldots. \tag{3.3}$$

The shifted Chebyshev polynomials are related to the better-known Chebyshev

polynomials $T_0, T_1, T_2, \ldots$ by the identity: $T_k^*(x) = T_k(2x-1)$. That is, they are obtained by shifting and scaling the Chebyshev polynomials to have domain $[0,1]$, see Mason and Handscomb (2002). We plot the first few shifted Chebyshev polynomials in Figure 3.1.

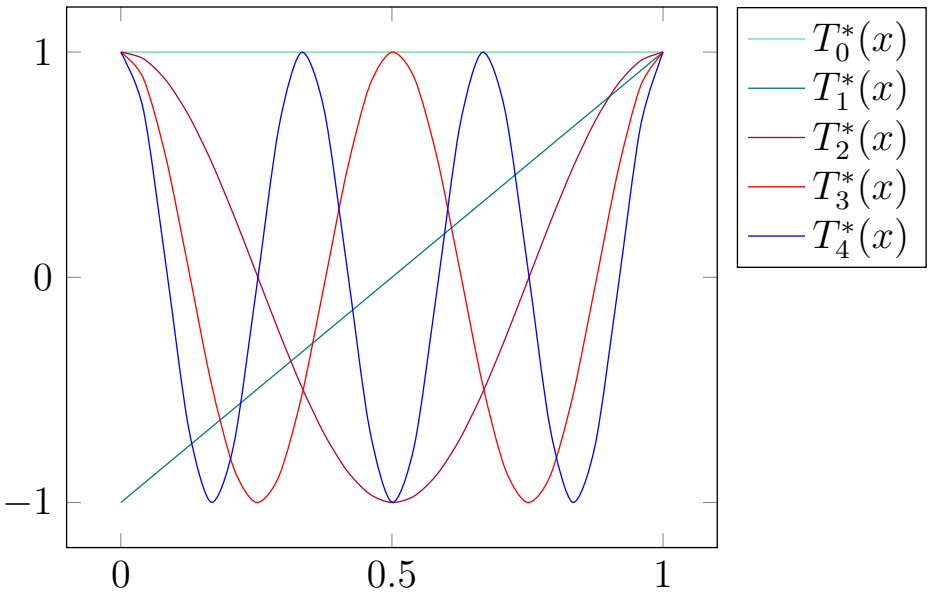


Figure 3.1: A plot of the first few shifted Chebyshev polynomials $T_0^*(x), \ldots, T_4^*(x)$.

Most sets of basis functions are orthogonal for some inner product. Shifted Chebyshev polynomials are an orthogonal basis with respect to the real inner product

$$\langle f, g \rangle = \int_0^1 f(x)g(x)w(x)dx$$

with weight function $w(x) = \frac{1}{\sqrt{x-x^2}}$. We have

$$\langle T_0^*, T_0^* \rangle = \int_0^1 T_0^*(x)T_k^*(x)w(x)dx = \pi, \tag{3.4}$$

$$\langle T_k^*, T_k^* \rangle = \int_0^1 T_k^*(x)T_k^*(x)w(x)dx = \frac{\pi}{2}, \qquad \text{for } k > 0 \tag{3.5}$$

$$\langle T_k, T_l \rangle = \int_0^1 T_k^*(x)T_l^*(x)w(x)dx = 0, \qquad \text{for } k \neq l. \tag{3.6}$$

See Mason and Handscomb (2002, pg.84) for details.

In order to express the backward diffusion in terms of shifted Chebyshev polynomials we use the relation between a first order monomial and a shifted Chebyshev

polynomial

$$xT_k^*(x) = \frac{1}{4}(T_{|k-1|}^*(x) + T_{k+1}^*(x)), \tag{3.7}$$

as well as the formula for the derivative

$$\frac{\partial T_k^*(x)}{\partial x} = \begin{cases} 4k \sum\limits_{\substack{r=1 \\ n-r \text{ odd}}}^{k-1} T_r^*(x) + 2kT_0^*(x), & \text{if } n \text{ is odd,} \\ 4k \sum\limits_{\substack{r=1 \\ n-r \text{ odd}}}^{k-1} T_r^*(x), & \text{if } n \text{ is even.} \end{cases} \tag{3.8}$$

Furthermore, the indefinite integral formula given by

$$\int T_k^*(x)dx = \begin{cases} \frac{1}{2}T_1^*(x) + C, \text{ for } k = 0, \\ \frac{1}{8}(T_2^*(x) - T_0^*(x)) + C, \text{ for } k = 1, \\ \frac{1}{4}\left(\frac{T_{k+1}^*(x)}{k+1} - \frac{T_{k-1}^*(x)}{k-1}\right) + C, \text{ for } k > 1. \end{cases} \tag{3.9}$$

See Mason and Handscomb (2002, pg.101) for derivations.

The expression for a Chebyshev polynomial as a monomial expansion is given by (Mason and Handscomb, 2002, Section 2.3.2 pg. 35)

$$T_k(x) = \sum_{m=0}^{\lfloor k/2 \rfloor} c_m^{(k)} x^{k-2m}$$

where

$$c_m^{(k)} = (-1)^m 2^{k-2m-1} \frac{k}{k-m} \binom{k-m}{m}.$$

We obtain an expression for the shifted Chebyshev polynomials by applying the Binomial theorem and the fact that $T_k^*(x) = T_k(2x-1)$. This leads to

$$T_k^*(x) = \sum_{m=0}^{\lfloor k/2 \rfloor} c_m^{(k)} \sum_{j=0}^{k-2m} (-1)^{k-2m-j} 2^j \binom{k-2m}{j} x^j. \tag{3.10}$$

However in practice we use a recursion formula to determine the monomial coefficients. It follows from the recursion formula (3.3) that T_k^* has degree k, so can be expressed

in terms of monomials

$$x^0, x^1 \ldots, x^k.$$

Denote the associated monomial coefficients

$$a_{k,0}, a_{k,1}, \ldots, a_{k,k},$$

so

$$T_k^*(x) = \sum_{i=0}^{k} a_{k,i} x^i.$$

Now in order to derive the recursion formula for monomial coefficients we use the recursion formula (3.3) and write T_k^* in terms of a general monomial expansion

$$\sum_{i=0}^{k+1} a_{k+1,i} x^i = 2(2x-1) \sum_{i=0}^{k} a_{k,i} x^i - \sum_{i=0}^{k-1} a_{k-1,i} x^i.$$

Further expanding this and rewriting subscripts we get

$$\sum_{i=0}^{k+1} a_{k+1,i} x^i = 4 \sum_{i=1}^{k+1} a_{k,i-1} x^i - 2 \sum_{i=0}^{k} a_{k,i} x^i - \sum_{i=0}^{k-1} a_{k-1,i} x^i.$$

Writing coefficients on the LHS in terms of coefficients on the RHS we get recursion formulas for the monomial coefficients as

$$a_{k+1,0} = -2a_{k,0} - a_{k-1,0}, \tag{3.11}$$

$$a_{k+1,i} = 4a_{k,i-1} - 2a_{k,i} - a_{k-1,i}, \qquad i = 1, \ldots, k, \tag{3.12}$$

$$a_{k+1,k+1} = 4a_{k,k}, \tag{3.13}$$

with $a_{0,0} = 1, a_{1,0} = -1$ and $a_{1,1} = 2$.

3.2.2 Expressing approximate partial likelihoods

There are two main ways of using shifted Chebyshev polynomials to approximate ℓ_i^B and ℓ_i^T as functions of x. The first is to approximate the function directly as a linear combination of the shifted Chebyshev polynomials, that is, finding sets of coefficients

$\lambda^B_{i,0}, \lambda^B_{i,1}, \lambda^B_{i,2}, \ldots$ and $\lambda^T_{i,0}, \lambda^T_{i,1}, \lambda^T_{i,2}, \ldots$ so that for all x in $[0,1]$ we have

$$\ell^B_i(x) \approx \sum_{k=0}^{K} \lambda^B_{i,k} T^*_k(x) \quad \text{and} \quad \ell^T_i(x) \approx \sum_{k=0}^{K} \lambda^T_{i,k} T^*_k(x).$$

It can be shown that the error in this approximation drops exponentially quickly as the number K of basis functions increases given that $\ell^B_i(x)$ and $\ell^T_i(x)$ is smooth, see Trefethen (2000, thm 1, pg. 30). We therefore only need to store a few coefficients in order to evaluate the partial likelihood function at any x, with small error.

The second way of obtaining an approximation is by determining the values $\ell^B_i(x)$ and $\ell^T_i(x)$ at a pre-specified set of points $x_0, x_1, \ldots, x_K$ and then finding the unique combination of coefficients $\lambda^B_{i,0}, \ldots, \lambda^B_{i,K}$ such that

$$\sum_{k=0}^{K} \lambda^B_{i,k} T^*_k(x_j) = \ell^B_i(x_j)$$

for all $j = 0, 1, \ldots, K$. This is called *polynomial interpolation*. As it happens, there is a particular choice of points $x_0, \ldots, x_K$ for which we can switch back and forth between function values

$$\ell^B_i(x_0), \ldots, \ell^B_i(x_K)$$

and interpolation coefficients

$$\lambda^B_{i,0}, \ldots, \lambda^B_{i,K}$$

with little numerical error and in $O(K \log K)$ time (Waldvogel, 2006; Trefethen, 2013). These points are called the Chebyshev-Lobatto points, defined by the rather opaque formula

$$x_j = \left(1 - \cos\left(\frac{2\pi j}{K}\right)\right) \Big/ 2, \text{ for } j = 0, \ldots, K.$$

These points are all in the interval $[0,1]$ with a denser packing of points nearer 0 and 1. More specifically, they are the projection of points spaced regularly around a unit semi-circle onto the interval $[0,1]$, see Figure 3.2.

In fact the Chebyshev-Lobatto points are the connection between Chebyshev polynomials and complex analysis (Trefethen, 2013). The choice of points allows the use

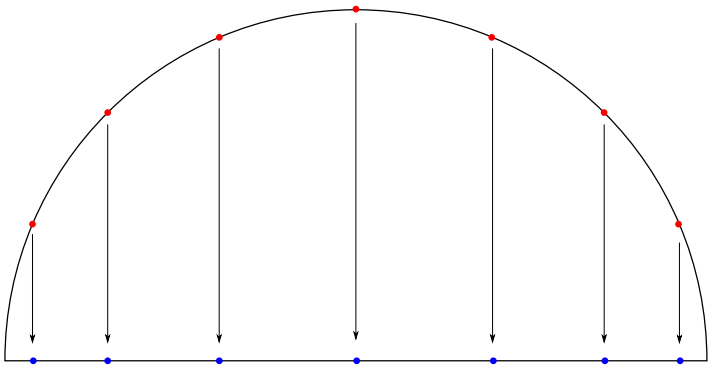


Figure 3.2: In order to get a Chebyshev-Lobatto grid we map equidistant points on a unit semi-circle onto the real line

of the discrete Fourier transform (DFT) to easily find the interpolation coefficients given the function values and vice versa. The DFT accepts $K+1$ equispaced points of a sampled function on a periodic grid, $[0, 2\pi]$ (in our case the function is sampled on the interval $[0, 1]$ and mapped to a periodic grid i.e semi-unit circle). It returns $K+1$ complex numbers which represents the amplitude and phase of a complex basis function. The amplitude is the coefficients of the shifted Chebyshev polynomials. See (Trefethen, 2013) for more detail. The same idea applies for the inverse DFT. In this case the inverse DFT takes interpolation coefficients as input and return function values evaluated at Chebyshev-Lobatto points. We use the fast Fourier transform algorithm (FFT) (Cooley and Tukey, 1965) to perform the calculation of the DFT and its inverse in $\mathcal{O}(K \log K)$ time. Note that both coefficient values $\lambda^B_{i,k}, \lambda^T_{i,k}$ and function values $\ell^B_i(x_j), \ell^T_i(x_j)$ are used when approximating the partial likelihoods ℓ^B_i and ℓ^T_i.

3.2.3 Clenshaw-Curtis type quadrature

When k is even, shifted Chebyshev polynomial T^*_k has the property that the definite integral on the interval $[0, 1]$ equals zero. When k is odd we have (Trefethen, 2013, pg. 192)

$$\int_0^1 T^*_k(x)dx = \frac{1}{1-k^2}. \tag{3.14}$$

Therefore approximate likelihoods

$$\ell^B_i(x) \approx \sum_{i=0}^{K} \lambda^B_{i,k} T^*_k(x)$$

can be integrated in $\mathcal{O}(K)$ time by taking the sum of the coefficients

$$\int_0^1 \sum_{k=0}^{K} \lambda_{i,k}^B T_k^*(x) dx = \sum_{k=0,k \text{ odd}}^{K} \frac{\lambda_{i,k}^B}{1-k^2}. \tag{3.15}$$

This is called *Clenshaw-Curtis type quadrature*.

3.3 Approximate partial likelihood at a leaf

We compute our approximate likelihood at the bottom of a leaf branch by simply evaluating the function

$$\ell_i^B(x_j) = \binom{2n_i}{r_i} x_j^{r_i}(1-x_j)^{2n_i-r_i} \text{ for } j = 0, 1 \ldots, K \tag{3.16}$$

at the Chebyshev-Lobatto points. We then compute corresponding coefficients

$$\lambda_{i,0}^B, \lambda_{i,1}^B, \ldots, \lambda_{i,K}^B$$

using the FFT algorithm described above.

3.4 Approximating the partial likelihood along a branch

Shifted Chebyshev polynomials provide the foundation for the numerical methods we use to solve the backward diffusion (1.9). Note that, unlike the forward diffusion (1.6), solutions to the backward diffusion with the right initial condition are smooth (infinitely differentiable) functions (see appendix for more on this), meaning that we can avoid some of the numerical headaches (such as solutions that tend toward infinite spikes) encountered by those using the forward diffusion directly (Lukic and Hey, 2012).

We approximate $g(x,t)$ in terms of shifted Chebyshev polynomials

$$g(x,t) \approx \sum_{k=0}^{K} \lambda_k(t) T_k^*(x). \tag{3.17}$$

Then

$$\frac{\partial g(x,t)}{\partial t} \approx \sum_{k=0}^{K} \frac{\partial \lambda_k(t)}{\partial t} T_k^*(x),, \tag{3.18}$$

$$\frac{\partial g(x,t)}{\partial x} \approx \sum_{k=0}^{K} \lambda_k(t) \frac{\partial T_k^*(x)}{\partial x} \tag{3.19}$$

and

$$\frac{\partial^2 g(x,t)}{\partial x^2} \approx \sum_{k=0}^{K} \lambda_k(t) \frac{\partial^2 T_k^*(x)}{\partial x^2}. \tag{3.20}$$

Let $\boldsymbol{\lambda}(t)$ denote the vector of values $\lambda_0(t), \ldots, \lambda_K(t)$.

Formulas for the derivative of shifted Chebyshev polynomials (3.8) lead eventually to a $(K+1) \times (K+1)$ matrix

$$\mathbf{D} = \begin{bmatrix} 0 & 2 & 0 & 6 & \cdots & 2N \\ & 0 & 8 & 0 & & \\ & & 0 & 12 & \ddots & \vdots \\ & & & \ddots & \ddots & 0 \\ & & & & 0 & 4N \\ & & & & & 0 \end{bmatrix}$$

such that

$$\sum_{k=0}^{K} \lambda_k(t) \frac{\partial T_k^*(x)}{\partial x} = \sum_{k=0}^{K} (\boldsymbol{D}\boldsymbol{\lambda}(t))_k T_k^*(x)$$

and

$$\sum_{k=0}^{K} \lambda_k(t) \frac{\partial^2 T_k^*(x)}{\partial x^2} = \sum_{k=0}^{K} (\boldsymbol{D}^2\boldsymbol{\lambda}(t))_k T_k^*(x).$$

Also, formulas for the product of a monomial and a shifted Chebyshev polynomial

(3.7) lead to a $(K+1)\times(K+1)$ matrix

$$\mathbf{M} = \begin{bmatrix} 0.5 & 0.25 & & & \\ 0.5 & 0.5 & \ddots & & \\ & 0.25 & \ddots & \ddots & \\ & & \ddots & \ddots & 0.25 \\ & & & 0.25 & 0.5 \end{bmatrix}$$

such that

$$\sum_{k=0}^{K} \lambda_k(t) x T_k^*(x) = \sum_{k=0}^{K} (\boldsymbol{M}\boldsymbol{\lambda}(t))_k T_k^*(x).$$

Indeed after some tedious algebra we can derive a $(K+1)\times(K+1)$ $\boldsymbol{Q}$ matrix so that

$$(\beta_1(1-x)-\beta_2 x)\frac{\partial}{\partial x}\sum_{k=0}^{K}\lambda_k(t)T_k^*(x)+\frac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2}\sum_{k=0}^{K}\lambda_k(t)T_k^*(x) = \sum_{k=0}^{K}(\boldsymbol{Q}\boldsymbol{\lambda}(t))_k T_k^*(x)$$

for all $x \in [0,1]$ with

$$\mathbf{Q} = (\beta_2(1-\mathbf{M}) - \beta_1\mathbf{M})\mathbf{D} + \frac{1}{N}\mathbf{M}(1-\mathbf{M})\mathbf{D^2}. \tag{3.21}$$

Plugging this and (3.18) into (1.9) we obtain an equation

$$\sum_{k=0}^{K}\frac{\partial\lambda_k(t)}{\partial t}T_k^*(x) = \sum_{k=0}^{K}(\boldsymbol{Q}\boldsymbol{\lambda}(t))_k T_k^*(x)$$

for an approximate solution to the PDE (1.9), giving the system

$$\frac{\partial\boldsymbol{\lambda}(t)}{\partial t} = \boldsymbol{Q}\boldsymbol{\lambda}(t). \tag{3.22}$$

with initial condition $\boldsymbol{\lambda}(0)$ given as the Chebyshev coefficients of $g(x,0)$.

Like Bryant *et al.* (2012) we use rational approximations to $\exp(\boldsymbol{Q}t)\boldsymbol{\lambda}(0)$ using a Caratheodory-Fejer approximation (more detail below). Additionally, we use techniques adapted from Fox and Parker (1968) (discussed in detail below) to take advantage of structure in the matrix $\mathbf{Q}$, allowing an implementation of the Caratheodory-

Fejer approximation which runs in $\mathcal{O}(K)$ time.

To summarise, consider a node i and let t_i denote the length (in coalescent units) of the branch connecting i to its parent. Suppose ℓ_i^B with corresponding coefficients $\lambda_{i,0}^B, \ldots, \lambda_{i,K}^B$ is already computed at the bottom of the branch. We then compute partial likelihood ℓ_i^T with corresponding coefficients $\lambda_{i,0}^T, \ldots, \lambda_{i,K}^T$ at the top of the branch by:

1. Setting $\boldsymbol{\lambda}(0) = \lambda_{i,0}^B, \ldots, \lambda_{i,K}^B$.
2. Computing a numerical approximation for $\exp(\boldsymbol{Q}t_i)\boldsymbol{\lambda}(0)$.
3. Setting $\lambda_{i,0}^T, \ldots, \lambda_{i,K}^T = \lambda(t_i)_0, \ldots, \lambda(t_i)_K$.

3.5 Linear time methods for solving diffusions approximately

3.5.1 Brief overview

Next we discuss how to set up a sparse matrix equivalent to $\mathbf{Q}$ which in turn will allow us to approximate $\exp \mathbf{Q}t$ in $\mathcal{O}(K)$ time!

As discussed the matrix $\boldsymbol{Q}$ is chosen so that if

$$g(x,t) = \sum_{k=0}^{K} \boldsymbol{\lambda}(t)_k T_k^*(x) \tag{3.23}$$

then

$$(\beta_1(1-x) - \beta_2 x)\frac{\partial}{\partial x} g(x,t) + \tfrac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2} g(x,t) \approx \sum_{k=0}^{K} (\mathbf{Q}\boldsymbol{\lambda}(t))_k T_k^*(x). \tag{3.24}$$

The bottleneck in the numerical method we use is the repeated solution of linear systems that look like

$$(\mathbf{Q} - z\mathbf{I})\mathbf{x} = \mathbf{v}$$

for complex values z and vectors $\mathbf{v}$. Using a direct method, these take $O(K^2)$ time each as $\mathbf{Q}$ is upper triangular. However we can do better, using a trick described in Fox and Parker (1968). Here we give a very high level description of the approach and flesh out the detail in sections that follow.

The key idea is to apply integration twice to the LHS of (3.24). Using integration by parts multiple times we have

$$\begin{aligned}\int_0^x \int_0^y \left[(\beta_1(1-z) - \beta_2 z)\frac{d}{dz}g(z,t) + \tfrac{1}{2}z(1-z)\frac{d^2}{dz^2}g(z,t)\right] dz\, dy \\ = -\int_0^x \int_0^y g(z,t)dz + (\beta_1(1-x) - \beta_2 x - 1 + 2x)\int_0^x g(z,t)dz \\ + \tfrac{1}{2}x(1-x)g(x,t) + (\tfrac{1}{2} - \beta_1)xg(0,t). \qquad (3.25)\end{aligned}$$

We introduce two new matrices $\mathbf{X}$ and $\mathbf{Y}$. The matrix $\mathbf{X}$ is derived from properties of shifted Chebyshev polynomials, and is defined so that if g is expanded as in (3.23) then

$$\int_0^x \int_0^y g(z)dz\, dy \approx \sum_{k=0}^{K} (\mathbf{X}\boldsymbol{\lambda}(t))_k T_k^*(x).$$

The matrix $\mathbf{Y}$ comes from (3.25) and has the property that if g is expanded as in (3.23) then the RHS of (3.25) equals

$$\sum_{k=0}^{K} (\mathbf{Y}\boldsymbol{\lambda}(t))_k T_k^*(x).$$

We then obtain

$$\mathbf{XQ} \approx \mathbf{Y}.$$

The usefulness of this follows from that fact that, with the exception of two rows, all the non-zero entries in $\mathbf{X}$ and $\mathbf{Y}$ are on or near the diagonal: both matrices are almost banded. To solve $(\mathbf{Q} - z\mathbf{I})\mathbf{x} = \mathbf{v}$ we can multiply both sides by $\mathbf{X}$ and solve the sparse system that results. In practise however to maintain the sparse structure of the system we solve a slightly different but still equivalent system. Overall, this now takes $O(K)$ time.

3.5.2 An explicit sparse expression for Y

The integration by parts (discussed above) show that there exists an equivalent LHS of (3.24) defined in terms of just $g(x,t)$ and not the derivatives of $g(x,t)$. The derivatives of $a(x) = \beta_2(1-x) - \beta_1 x$ and $b(x) = \frac{1}{2}x(1-x)$ are used instead.

To find the matrix $\boldsymbol{Y}$ we use formulas for the indefinite integral (3.9) to first set-up a $(K+1)\times(K+1)$ matrix

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 & \cdots & & 0 \\ 1/2 & 0 & -1/4 & & & \\ & 1/8 & 0 & -1/8 & & \\ & & 1/12 & 0 & \ddots & \\ & & & \ddots & \ddots & -\frac{1}{2N} \\ & & & & \frac{1}{2(N+1)} & 0 \end{bmatrix}.$$

Then the associated matrix $\mathbf{Y}$ is computed by simply applying $\boldsymbol{S}$ twice, that is $\mathbf{S^2Q} \approx \mathbf{Y}$. As long as we can express $a(x)$ and $b(x)$ and derivatives sparsely the resulting matrix will be sparse. Therefore we introduce the matrix $\mathbf{X} = \mathbf{S^2}$ that will help solve the backward diffusion efficiently. However to find $\mathbf{Y}$ this way takes $\mathcal{O}(K^3)$ time. There are faster ways to do this in $\mathcal{O}(K^2)$ time using recursions, see Trefethen (2000, pg. 61). This is still not good enough.

We therefore use the shifted Chebyshev identities (3.7), (3.8) and (3.9) to derive most of the explicit expressions required to set up $\boldsymbol{Y}$.

Some really really tedious algebraic manipulations and derivations was used to show (as expected) that the identities (3.8) and (3.9) cancel out when both are applied to T_k^* (see Appendix C for details). We then used this result throughout the rest of the derivations to build an ensemble of expressions (Lemma 3.27). These expressions are strung together in Theorem 3.29 to find the explicit expression for $\boldsymbol{Y}$. All the derivations follow the same recipe (See Appendix C for details). After some straightforward manipulation and grouping of coefficients in a step-by-step sort of way we eventually find that most terms cancel out nicely with a few terms close to the diagonal remaining (as expected). This exercise have two important consequences. First it proves that our matrix $\mathbf{Y}$ is always sparse. Secondly it allows us to construct $\mathbf{Y}$ in $\mathcal{O}(K)$ time. Note that we drop the integration constant in the lemma below.

Lemma 3.5.1. *We can express the double integral operators with monomials and shifted Chebyshev derivatives explicitly, in terms of only shifted Chebyshev polynomials,*

as:

$$\int\int x\frac{\partial}{\partial x}T_k^*(x)dxdy = -\frac{k}{16(k-1)(k-2)}T_{k-2}^* - \frac{1}{8(k-1)}T_{k-1}^* + \frac{1}{8\left(k^2-1\right)}T_k^* + \frac{1}{8(k+1)}T_{k+1}^*\frac{n}{16(k+1)(k+2)}T_{k+2}^*, \text{ for } k>1. \quad (3.26)$$

$$\int\int x\frac{\partial^2}{\partial x^2}T_k^*(x)dxdy = \frac{k+1}{4(k-1)}T_{k-1}^*(x)+\frac{3}{8}T_k^*(x)+\frac{k-1}{4(k+1)}T_{k+1}^*(x), \text{ for } k>2. \quad (3.27)$$

$$\int\int x^2\frac{\partial^2}{\partial x^2}T_k^*(x)dxdy = 2kT_1^*(x) + \frac{k(k+1)}{16\left(k^2-3k+2\right)}T_{k-2}^*(x) + \frac{k+1}{4(k-1)}T_{k-1}^*(x)+ \frac{5-3k^2}{8-8k^2}T_k^*(x) + \frac{k-1}{4(k+1)}T_{k+1}^*(x) + \frac{(k-1)k}{16(k+1)(k+2)}T_{k+2}^*(x), \text{ for } k>1. \quad (3.28)$$

Proof. See Appendix C for derivations. □

Theorem 3.5.2. *We can express the matrix* $\mathbf{Y}$ *explicitly, in terms of only shifted Chebyshev polynomials, as:*

$$\begin{aligned}&\int\int(\beta_1(1-x)-\beta_2x)\frac{\partial}{\partial x}T_k^*(x)+\frac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2}T_k^*(x)dxdy\\ &=\frac{-k(\beta_1+\beta_2-2(k+1))}{16(k-2)(k-1)}T_{k-2}^*(x)+\frac{\beta_2-\beta_1}{8(k-1)}T_{k-1}^*(x)+\frac{4-3k^2}{8(1-k^2)}T_k^*(x)\\ &+\frac{\beta_2-\beta_1}{8(k+1)}T_{k-1}^*(x)-\frac{k(\beta_1+\beta_2-2(k-1))}{16(k+2)(k+1)}T_{k+2}^*(x)+C_0T_0(x)+C_1T_1(x), \quad \text{for } k>1.\end{aligned} \quad (3.29)$$

Proof. Now the explicit expression follows from equations (3.26), (3.27) and (3.28) in Lemma 3.5.1. Note that C_0 and C_1 follow from the integration constants in the formula (3.9). □

To solve an equivalent system to $(\mathbf{Q}-z\mathbf{I})\mathbf{x}=\mathbf{v}$ uniquely and to avoid dealing with integration constants we find the last two explicit terms of

$$(\beta_1(1-x)-\beta_2x)\frac{\partial}{\partial x}T_k^*(x)+\frac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2}T_k^*(x). \quad (3.30)$$

We follow the same steps as the derivations in Appendix C to find

$$(\beta_1(1-x)-\beta_2 x)\frac{\partial}{\partial x}T_k^*(x)+\frac{1}{2}x(1-x)\frac{\partial^2}{\partial x^2}T_k^*(x)$$
$$=-(k^2-(k+\beta_1+\beta_2))T_k^*(x)+(\beta_1-\beta_2)kT_{k-1}^*(x)+\ldots, \quad \text{for } k>2.. \quad (3.31)$$

This will come in handy in the next section when we set up a sparse solver.

3.5.3 Approximating the matrix exponential

The key to solving the backward diffusion along a branch is approximating

$$\exp(\boldsymbol{Q}t)\boldsymbol{\lambda}(0)=\left(\sum_{k=0}^{\infty}\frac{(\boldsymbol{Q}t)^k}{k!}\right)\boldsymbol{\lambda}(0).$$

Schmelzer and Trefethen (2007) describe a method that uses the Caratheodory-Fejer procedure to compute rational approximations to $f(\mathbf{Q})=\exp\mathbf{Q}t$. Here we outline the mechanics of the procedure but to see why it works is quite involved and we refer to Trefethen (2013, Chapter20) for the theory behind the procedure. The depth of this approximation method is nicely emphasised with the following excerpt: "The conceptual and theoretical significance of the technique, however, goes beyond this. Indeed, the eigenvalue/singular value analysis of Caratheodory-Fejer approximation seems to be the principal known algebraic window into the detailed analysis of best approximations..." (Trefethen, 2013, pg. 211).

The standard implementation of the Caratheodory-Fejer method involves solving an iteration of shifted systems at common poles. For each pole $i=1,\ldots,p$ we introduce $c_i, z_i \in \mathbb{C}$. The shifted system is defined as

$$(\mathbf{Q}t-z_i\mathbf{I})\boldsymbol{w}_i=\boldsymbol{\lambda}(0). \quad (3.32)$$

It then follows that

$$\boldsymbol{\lambda}(t)=\sum_{i=1}^{p}\mathrm{Re}(c_i\boldsymbol{w}_i), \qquad \text{for } p=1,\ldots,12,$$

is a machine precision approximation of

$$\exp(\mathbf{Q}t)\boldsymbol{\lambda}(0).$$

Algorithm 5 Caratheodory-Fejer procedure

procedure EXP($\mathbf{Y}, \mathbf{X}, \boldsymbol{\lambda}(0)$)
  Set $\boldsymbol{v} \leftarrow \boldsymbol{\lambda}(0)$
  for $k = 0$ to number of timesteps n **do**
    for $i = 1$ to number of common poles p **do**
      Solve $(\mathbf{Y} - z_i\mathbf{X}/\Delta t)\boldsymbol{w}_i = \mathbf{X}\boldsymbol{v}/\Delta t$
    end for
    Set $\boldsymbol{v} \leftarrow \sum_{i=1}^{p} \mathrm{Re}(\boldsymbol{w}_i c_i)$
  end for
  return $\boldsymbol{\lambda}(t) = \sum_{i=1}^{p} \mathrm{Re}(\boldsymbol{w}_i c_i)$
end procedure

Algorithm for solving backward diffusion along a branch with initial condition $\boldsymbol{\lambda}(0)$.

We listed the values used for common poles c_i and z_i for $i = 1, \ldots, 12$ in Table 3.1 (Schmelzer and Trefethen, 2007). For large $t > 0$ the set of systems is solved via a timestep iteration

$$(\mathbf{Q} - z_i\mathbf{I}/\Delta t)\boldsymbol{w}_i = \boldsymbol{v}_t^*/\Delta t, \tag{3.33}$$

where $\boldsymbol{v}_{t^*+\Delta t} = \sum_i \mathrm{Re}(c_i\boldsymbol{w}_i)$ and $\boldsymbol{v}_0 = \boldsymbol{\lambda}(0)$.

However by multiplying the system (3.33) with the matrix operator $\boldsymbol{X}$ (in our case $\boldsymbol{S}^2$) we can instead solve the equivalent sparse system

$$(\mathbf{Y} - z_i\mathbf{X}/\Delta t)\boldsymbol{w}_i = \mathbf{X}\boldsymbol{v}_t^*/\Delta t, \tag{3.34}$$

where $\boldsymbol{v}_{t^*+\Delta t} = \sum_i \mathrm{Re}(c_i\boldsymbol{w}_1)$ and $\boldsymbol{v}_0 = \mathbf{X}\boldsymbol{\lambda}(0)$. To solve this system only takes $O(K)$ time instead of the previously required $O(K^2)$ time (since the operator $\mathbf{Q}$ is upper-diagonal), see Golub and Van Loan (2013, pg. 176) for methods to solve banded systems. In Algorithm 5 we describe the Caratheodory-Fejer procedure.

3.5.4 Setting up a multi-thread ODE solver

When mutation rates (β_1, β_2) are equal we can solve the system

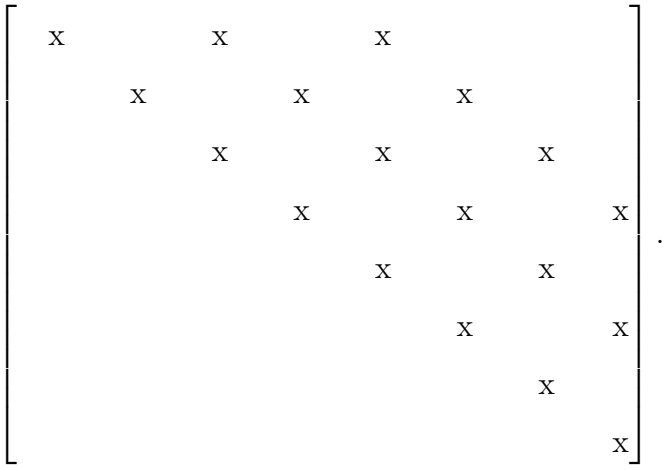
$$(\mathbf{Y} - z\mathbf{X})w = \mathbf{v}$$

Poles	c_i		
1	0.000818433612497	+	0.000581353207069j
2	-0.068571505514864	-	0.038419074245887j
3	1.319411815998137	+	0.183523497750480j
4	-8.238258033274786	+	2.796192505614474j
5	18.785982629476070	-	20.237292093573895j
6	-11.799383335697918	+	46.411650777279597j
7	-11.799383335697890	-	46.411650777279569j
8	18.785982629476067	+	20.237292093573895j
9	-8.238258033274763	-	2.796192505614448j
10	1.319411815998138	-	0.183523497750480j
11	-0.068571505514865	+	0.038419074245888j
12	0.000818433612497	-	0.000581353207069j
	z_i		
1	-6.998688082445778	-	13.995917029301355j
2	-2.235968223749446	-	11.109296400461870j
3	0.851707264834878	-	8.503832905826961j
4	2.917868800307170	-	6.017345968518187j
5	4.206124506834328	-	3.590920783130140j
6	4.827493775040721	-	1.193987999180278j
7	4.827493775040721	+	1.193987999180278j
8	4.206124506834328	+	3.590920783130140j
9	2.917868800307170	+	6.017345968518187j
10	0.851707264834878	+	8.503832905826961j
11	-2.235968223749446	+	11.109296400461870j
12	-6.998688082445778	+	13.995917029301355j

Table 3.1: List of values for common poles $i = 1, \ldots, 12$ used in the Caratheodory-Fejer procedure (Trefethen, 2013).

in parallel. In particular we set up a sparse solver based on two smaller banded upper-diagonal systems of roughly half the size of $\boldsymbol{Y}$.

It follows from (3.29) and (3.31) that when $\beta_1 = \beta_2$ we solve a sparse system with the following structure



This implies that even and uneven rows are mostly independent. Therefore we can split the even and uneven rows into two separate systems (see Figure 3.3). These systems can be solved independently of one another (on two separate threads in parallel). We summarise this procedure in Algorithm 6.

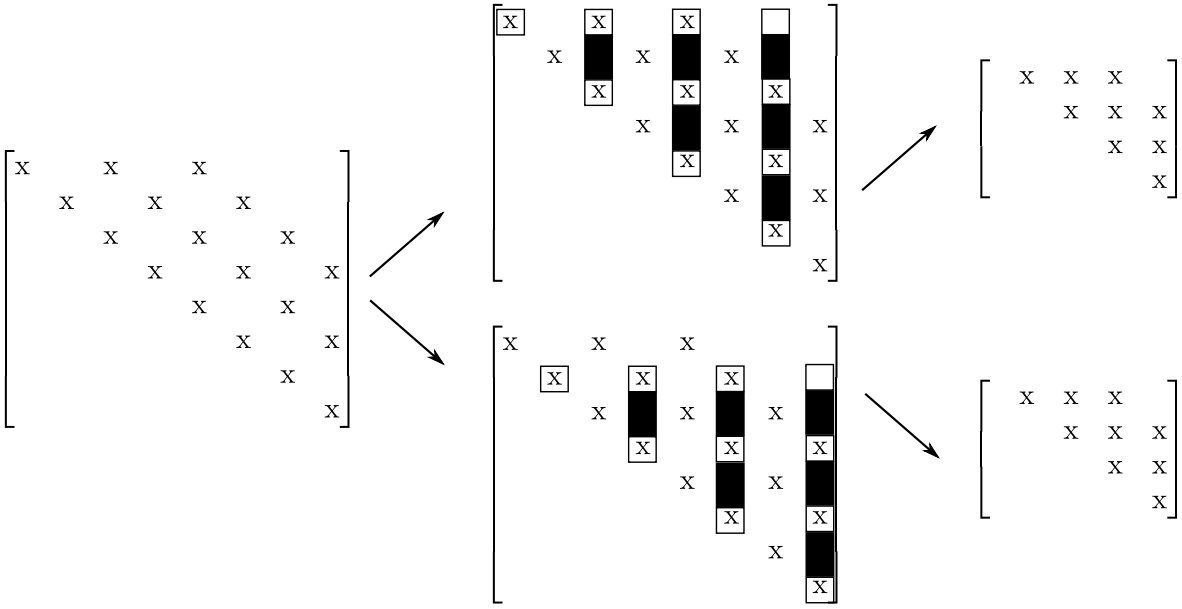


Figure 3.3: We can split the matrix operator $(\mathbf{Y} - \boldsymbol{z}\mathbf{X})$ into two smaller upper-diagonal matrices by sorting the even and uneven rows of $(\mathbf{Y} - \boldsymbol{z}\mathbf{X})$ into separate matrices.

Algorithm 6 ODE Solver for equal mutation rates

1: **procedure** SOLVE($\mathbf{Y}, \mathbf{X}, v$)
2: Split $(\mathbf{Y} - \boldsymbol{z}\mathbf{X})$ into two smaller matrices $\boldsymbol{S}_e$ and $\boldsymbol{S}_u$, see Figure 3.3 for more detail.
3: Separate even rows ($\mathbf{X}\boldsymbol{v}_e$) and uneven rows ($\mathbf{X}\boldsymbol{v}_u$) of $\mathbf{X}\boldsymbol{v}$
4: Solve $\boldsymbol{S}_e w_e = v_e$ and $\boldsymbol{S}_u w_u = v_u$ using backward substitution
5: Get $\boldsymbol{w}$ by splicing $\boldsymbol{w}_e$ and $\boldsymbol{w}_u$ together
return $\boldsymbol{w}$
6: **end procedure**

A parallel ODE solver. Note that line 4 can be executed on multiple threads.

3.6 Approximating partial likelihoods at a speciation

Consider a parent node i with two children nodes j and l. Given approximations

$$\ell_j^T(x) \approx \sum_{k=0}^{K} \lambda_{j,k} T_k(x),$$

and

$$\ell_l^T(x) \approx \sum_{k=0}^{K} \lambda_{l,k} T_k(x),$$

we can evaluate $\ell_j^T(x)$ and $\ell_l^T(x)$ at the Chebyshev-Lobatto points $(x_0, x_1, \ldots, x_K)$ in $\mathcal{O}(K \log K)$ time using the FFT algorithm (as discussed in Section 3.2.2). We then compute an approximation for the partial likelihood $\ell_i^B(x)$ as

$$\ell_i^B(x_p) = \ell_j^T(x_p)\ell_l^T(x_p) \text{ for } p = 0, 1, \ldots, K. \tag{3.35}$$

This computation takes $O(K)$ time.

3.7 Approximating likelihoods at the root

3.7.1 Description of the numerical problem

So far we have shown efficient ways to compute partial likelihoods at the leaves of a tree all the way up to the root. There is however one more challenging computation that remains, the integration at the root to compute the likelihood

$$L_m = \int_0^1 \pi(x|\beta_1, \beta_2)\ell_\rho^B(x)dx, \tag{3.36}$$

see (2.9). The reason for difficulty in integrating the likelihood at the root is due to the shape of the stationary density (2.1) when mutation rates (β_1, β_2) are low. In particular for $\beta_1, \beta_2 < 1$ the beta distribution has singularities at the boundaries which follow from the term

$$x^{\beta_1 - 1}, \qquad 0 < \beta_1 < 1$$

and the term

$$(1-x)^{\beta_2 - 1}, \qquad 0 < \beta_2 < 1$$

in the beta density (2.1). Typical fast numerical integration techniques such as quadrature methods will give poor approximations even when intervals are reasonably small. Even Clenshaw-Curtis quadrature will fail since polynomial functions (even of high degree) cannot approximate functions with singularities very well. We investigated the following three methods:

- The first method uses monomial formulas for the partial likelihood $\ell_\rho^B(x)$ to express L_m in terms of moments of the beta distribution.
- The second method analytically manipulates the Chebyshev basis functions to express the partial likelihood $\ell_\rho^B(x)$ in a way that eliminates singularities due to the beta density (2.1).
- The third method uses a mean approximation for the partial likelihood $\ell_\rho^B(x)$ and integrates intervals of the beta density via a continued fraction expansion.

3.7.2 Method 1: Integration via moments of the Beta distribution

We remind ourselves that we can write T_k^* in terms of the monomials

$$x^0, x^1 \ldots, x^k$$

with the associated monomial coefficients

$$a_{k,0}, a_{k,1}, \ldots, a_{k,k}$$

as

$$T_k^*(x) = \sum_{i=0}^{k} a_{k,i} x^i.$$

The i-th moment of the beta distribution $\pi(x|\beta_1, \beta_2)$ for $\beta_2 > 0$ and $2\beta_1 + i > 0$ equals

$$E(x^i) = \int_0^1 x^i \pi(x|\beta_1, \beta_2) dx \tag{3.37}$$

$$= \frac{\Gamma(2\beta_2)\,\Gamma(i + 2\beta_1)}{B(2\beta_1, 2\beta_2)\,\Gamma(i + 2\beta_1 + 2\beta_2)}, \tag{3.38}$$

see Forbes *et al.* (2011, pg. 55). We substitute the Chebyshev expansion of $\ell_\rho^B(x)$ (refer to Chapter 2 for context)

$$\sum_{k=0}^{K} \lambda_{\rho,k}^B T_k^*(x) dx$$

into the partial likelihood L_m to get

$$\int_0^1 \pi(x|\beta_1, \beta_2) x^{2\beta_2 - 1} (1 - x)^{2\beta_1 - 1} \sum_{k=0}^{K} \lambda_{\rho,k}^B T_k^*(x) dx.$$

Then we substitute the monomial expansion of T_k^* for $k = 0, \ldots, K$ into L_m

$$\int_0^1 \pi(x|\beta_1, \beta_2)(\lambda_{\rho,0}^B a_{0,0} + \lambda_{\rho,1}^B (a_{1,0} + a_{1,1}x) + \cdots + \lambda_{\rho,K}^B \left(a_{K,0} + a_{K,1}x + \cdots + a_{K,K}x^K\right)) dx.$$

Rearranging terms we get

$$\int_0^1 \pi(x|\beta_1,\beta_2)\left((\lambda^B_{\rho,0}a_{0,0}+\cdots+\lambda^B_{\rho,K}a_{K,0})+ (\lambda^B_{\rho,0}a_{0,1}+\cdots+\lambda^B_{\rho,K-1}a_{(K-1),1})x+\cdots+\lambda^B_{\rho,K}a_{K,K}x^K\right)dx.$$

Lastly we substitute the moments of the beta distribution $E(x^i)$ for $i = 0,\ldots,K$ to get

$$L_s = \sum_{k=0}^{K}\lambda^B_{\rho,0}a_{k,0} + \sum_{k=0}^{K-1}\lambda^B_{\rho,0}a_{k,1}E(x)+\cdots+\lambda^B_{\rho,K}a_{K,K}E(x^K).$$

Now in order to compute L_m efficiently we determine the monomial coefficients via the recursion formulas (3.11),(3.12),(3.13) (these can be precomputed) and calculate the moments of the beta density using the explicit expression . This method takes $\mathcal{O}(K^2)$ time since we have to add (and multiply) terms for each moment of the beta density.

3.7.3 Method 2: Integration via separation and Clenshaw-Curtis quadrature

In this approach we separate out those parts which are difficult to integrate numerically and determine them analytically. Suppose $f(x)$ is a Chebyshev polynomial approximation of $\ell^B_\rho(x)$

$$f(x) = \sum_{k=0}^{K}\lambda^B_{\rho,k}T^*_k(x) \approx \ell^B_\rho(x).$$

We factor this polynomial as

$$f(x) = x(1-x)g(x) + f(0) + (f(1)-f(0))x$$

and then compute

$$\int_0^1 f(x)\pi(x|\beta_1,\beta_2)dx = \int_0^1 g(x)x(1-x)\pi(x|\beta_1,\beta_2)dx + \int_0^1 (f(0) + (f(1)-f(0))x)\pi(x|\beta_1,\beta_2)dx. \quad (3.39)$$

Noting that the first integral is now well-behaved while the second integral evaluates

to $f(0) + (f(1) - f(0))\frac{\beta_1}{\beta_1+\beta_2}$ by properties of the beta distribution.

Now in order to evaluate the first integral on the RHS of (3.39) efficiently we have to find

$$g(x) = \sum_{k=0}^{K} \lambda^g_{\rho,k} T^*_k(x)$$

such that

$$\sum_{k=0}^{K} \lambda^B_{\rho,k} T^*_k(x) = x(1-x) \sum_{k=0}^{K} \lambda^g_{\rho,k} T^*_k(x) + \int_0^1 (f(0)+(f(1)-f(0))x)\pi(x|\beta_1,\beta_2)dx \quad (3.40)$$

is satisfied.

To find the coefficients $\lambda^g_{\rho,k}$, for $k = 0, \ldots, K$ we substitute

$$x(1-x) = \frac{1}{8} - \frac{1}{8}T^*_2$$

into the equation (3.40) to get

$$\sum_{k=0}^{K} \lambda^B_{\rho,k} T^*_k(x) = \frac{1}{8}\left(\sum_{k=0}^{K-2} \lambda^g_{\rho,k} T^*_k(x)\right) - \frac{1}{8}(\sum_{k=0}^{K-2} \lambda^g_{\rho,k} T^*_2(x) T^*_k(x)) + f(0) + \frac{\beta_1}{\beta_1+\beta_2}(f(1)-f(0)). \quad (3.41)$$

Now applying

$$T^*_2(x)T^*_k(x) = \frac{1}{2}\left(T^*_{k-2}(x) + T^*_{k+2}(x)\right)$$

and carefully rearranging terms we have

$$\sum_{k=0}^{K} \lambda_{\rho,k}^{B} T_k^*(x) = \left(f(0) + \frac{\beta_1}{\beta_1+\beta_2}(f(1)-f(0)) + \frac{1}{8}\lambda_{\rho,0}^{g} - \frac{1}{16}\lambda_{\rho,2}^{g} \right) T_0^*(x)$$
$$+ \left(\frac{1}{16}\lambda_{\rho,1}^{g} + \frac{1}{8}\lambda_{\rho,3}^{g} \right) T_1^*(x) + \left(\frac{1}{8}\lambda_{\rho,2}^{g} - \frac{1}{8}\lambda_{\rho,0}^{g} - \frac{1}{16}\lambda_{\rho,4}^{g} \right) T_2^*(x)$$
$$+ \left(\frac{1}{8}\lambda_{\rho,3}^{g} - \frac{1}{16}\lambda_{\rho,1}^{g} - \frac{1}{16}\lambda_{\rho,5}^{g} \right) T_3^*(x) + \dots$$
$$+ \left(\frac{1}{8}\lambda_{\rho,N-4}^{g} - \frac{1}{16}\lambda_{\rho,N-6}^{g} - \frac{1}{16}\lambda_{\rho,N-2}^{g} \right) T_{N-4}^*(x)$$
$$+ \left(\frac{1}{8}\lambda_{\rho,N-3}^{g} - \frac{1}{16}\lambda_{\rho,N-5}^{g} \right) T_{N-3}^*(x) + \left(\frac{1}{8}\lambda_{\rho,N-2}^{g} - \frac{1}{16}\lambda_{\rho,N-4}^{g} \right) T_{N-2}^*(x)$$
$$- \left(\frac{1}{16}\lambda_{\rho,N-3}^{g} \right) T_{N-1}^*(x) - \left(\frac{1}{16}\lambda_{\rho,N-2}^{g} \right) T_N^*(x). \quad (3.42)$$

Now using the above expression for the coefficients we can write the shifted Chebyshev coefficients in terms of a banded upper diagonal matrix $\boldsymbol{A}$ such that

$$\mathbf{A}\boldsymbol{\lambda}_{\rho}^{g} = \boldsymbol{\lambda}_{\rho}^{B}. \quad (3.43)$$

This system can be solved in $\mathcal{O}(K)$ time using backwards substitution. Once we have the coefficients $\boldsymbol{\lambda}_{\rho}^{g}$ we can evaluate the integral

$$\int_0^1 g(x)x(1-x)\pi(x|\beta_1,\beta_2)dx$$

using Clenshaw-Curtis quadrature and the DFT in $\mathcal{O}(K \log K)$ time.

We summarize the method in Algorithm 7.

3.7.4 Method 3: Integration via the Incomplete beta function

The *Incomplete beta* function is defined as

$$\int_0^x \frac{x^{\beta_1-1}(1-x)^{\beta_2-1}}{\mathcal{B}(\beta_1,\beta_2)}dx.$$

Using a modified Lents's method (Press *et al.*, 2007, pg. 206) it can be evaluated efficiently as a continued fraction expansion (Press *et al.*, 2007, pg. 270). See Algorithm 8 for details. This expansion converges rapidly when β_1 and β_2 are small. Therefore by

Algorithm 7 Method 2

procedure INTEGRATE($\sum_{j=0}^{K} \lambda_{\rho,j}^{B}, \beta_1, \beta_2$)
    Set up $\boldsymbol{A}$ using the expressions (3.42)
    Solve the linear system (3.43) to get coefficients $\boldsymbol{\lambda}_\rho^g$
    Switch to values $\boldsymbol{\ell}_\rho^g$ from coefficients $\boldsymbol{\lambda}_\rho^g$
    Compute values $\ell_\rho^\pi(x_j)$ by evaluating $x_j(1-x_j)\pi(x_j|\beta_1,\beta_2)$ for $j = 0, \ldots, K$.
    Compute $\ell_\rho(x_j)$ using $\ell_\rho^g(x_j), \ell_\rho^\pi(x_j)$ for $j = 0, \ldots, K$,
    Switch to coefficients $\boldsymbol{\lambda}_\rho$ from values $\boldsymbol{\ell}_\rho(x_j)$
    Compute 1st term in LHS of (3.39) using coefficients $\boldsymbol{\lambda}_\rho$ and Clenshaw-Curtis quadrature.
    Evaluate 2nd term in LHS of (3.39)
    return $log(L_s)$
end procedure

approximating the partial likelihood $\ell_\rho^B(x)$ on the interval $[x_k, x_{k+1}]$ by the mean value

$$\frac{\ell_\rho^B(x_{k+1}) - \ell_\rho^B(x_k)}{2}$$

we can write

$$\int_0^1 \ell_\rho^B(x)\pi(x|\beta_1,\beta_2)dx \approx \sum_{k=0}^{K-1} \frac{\ell_\rho^B(x_{k+1}) - \ell_\rho^B(x_k)}{2} \int_{x_i}^{x_{k+1}} \pi(x|\beta_1,\beta_2)dx.$$

Now we can evaluate the integral using the Incomplete beta function and the modified Lent's method. To see this we note that

$$\int_{x_k}^{x_{k+1}} \pi(x|\beta_1,\beta_2)dx = \int_0^{x_{k+1}} \pi(x|\beta_1,\beta_2)dx - \int_0^{x_k} \pi(x|\beta_1,\beta_2)dx.$$

Also we can efficiently evaluate $\ell_\rho^B(x_k)$ for $k = 0, \ldots, K$ using the FFT algorithm. We precompute the Beta density integral on the intervals $[x_k, x_{k+1}], k = 0, \ldots, K$. This allows us to evaluate the likelihood at the root L_m in $\mathcal{O}(K \log K)$ time.

3.7.5 Comparing numerical integration methods

Protocol

In this numerical experiment we compare the rate of convergence for three integration methods discussed above. The accuracy of all three of the methods are determined

Algorithm 8 Method 3

1: **procedure** CFRACTAL(x, β_1, β_2, e)
2: $m \leftarrow 1$
3: **while** $\|\delta - 1\| \leq e$ **do**
4: $a \leftarrow (m(\beta_2 - m))/((\beta_1 - 1 + 2m)(\beta_1 + 2m))x$
5: $b \leftarrow 1/(1 + a \cdot b)$
6: $c \leftarrow 1 + a/c$
7: $h \leftarrow b \cdot c \cdot h$
8: $a \leftarrow (-(\beta_1 + m)(\beta_1 + \beta_2 + m))/((\beta_1 + 2m)(\beta_1 + 1 + 2m))x$
9: $b \leftarrow 1/(1 + a \cdot b)$
10: $c \leftarrow 1 + a/c$
11: $\delta \leftarrow d \cdot c$
12: $h \leftarrow h \cdot \delta$
13: $m \leftarrow m + 1$
14: **end while**
return h
15: **end procedure**
16: **procedure** INCOMPLETE BETA(x, β_1, β_2)
17: $b_1 \leftarrow \exp(\ln \Gamma(\beta_1 + \beta_2) - \ln \Gamma(\beta_1) - \ln \Gamma(\beta_2) + \beta_1 \log(x) + \beta_2 \log(1 - x))$
18: **if** $x < (\beta_1 + 1)/(\beta_1 + \beta_2 + 2)$ **then**
19: $\mathcal{B} \leftarrow b_1$CFRACTAL$(x, \beta_1, \beta_2, e)/\beta_1$
20: **else**
21: $\mathcal{B} \leftarrow 1 - b_1$CFRACTAL$(x, \beta_1, \beta_2, e)/\beta_2$
22: **end if**
return $\mathcal{B}$
23: **end procedure**

We give the algorithm for evaluating the incomplete beta function as a continued fraction expansion (Press *et al.*, 2007, pg. 270).
Note for the first procedure CFRACTAL we also specify the approximation error e.

by the number of Chebyshev basis functions (K) used in approximations. We want to know which method converges to the true values the fastest as well as assess the numerical stability.

We generated five 4 taxa species trees using the Yule distribution with speciation rate of 10. We generated population size (θ) parameters from a Gamma distribution with mean 0.01 and variance 0.0001. We then simulated data for one site for each tree (with 32 total individuals) under the diffusion model.

We then computed the log-likelihood for number of basis function, $K = 9, 14, 19 \ldots, 49$. Note that for this experiment we do not limit number of Chebyshev basis functions (K) to values of the form $2^m + 1$. Since there is no analytical expression for the likelihood we assess convergence by comparing values calculated to those with a large ($K = 100$) number of Chebyshev basis functions. Furthermore integration at the root for large number of basis functions ($K = 100$) we use a robust integration method, namely QAGS. QAGS is an adaptive integration method based on 21-point Gauss–Kronrod quadrature from the FORTRAN library (Piessens *et al.*, 2012). In order to study convergence we compute relative error by increasing the number of basis functions $K = 9, 14, 19 \ldots, 49$ for each method.

Results

We summarise the convergence results of the three methods in Figure 3.4. We see that method 3 is slowest to converge. Typically method 1 and method 2 have the same relative error when K is small. However method 1 becomes unstable when the number of basis functions gets large enough. This is due to the monomial coefficients that blow up. We can see why by looking at the monomial coefficient formula (3.10). The results indicate that method 2 is stable and is fastest to converge. Therefore we implemented method 2 in Snapper.

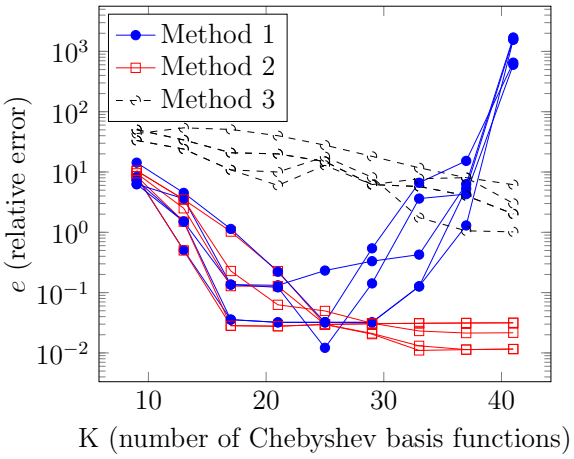


Figure 3.4: Relative error comparison between numerical integration methods at the root of a 4 taxa tree. Method 1 use moments of a beta distribution; Method 2 analytically manipulates Chebyshev basis function and then apply Clenshaw-Curtis integration. Method 3 use a mean approximation of the partial likelihood and evaluates intervals of the beta density.

3.8 Computing the log-likelihood of a species trees

We briefly review the algorithmic steps for computing the likelihood of a species tree at a particular site (see Figure 3.5 for illustration of below steps).

1. At a leaf: Fit the initial condition of the backward diffusion at a leaf node from a given sample, that is:

$$g(x,0) = \binom{n}{r} x^r (1-x)^{n-r} \tag{3.44}$$

   where n is the sample size and r is the number of observed red alleles.

2. Given the partial likelihood at the bottom of a branch, we compute the partial likelihood at the top of the branch: Solve

$$\frac{\partial g(x,t)}{\partial t} = (\beta_2(1-x) - \beta_1 x)\frac{\partial}{\partial x} g(x,t) + \frac{1}{2}x(1-x)\frac{1}{2}\frac{\partial^2}{\partial x^2} g(x,t). \tag{3.45}$$

   starting at the bottom of the branch until the top of the branch is reached.

3. Multiply the partial likelihoods at a speciation.

4. The likelihood of a site is then calculated by integrating over the root partial likelihood and the stationary distribution (2.1).

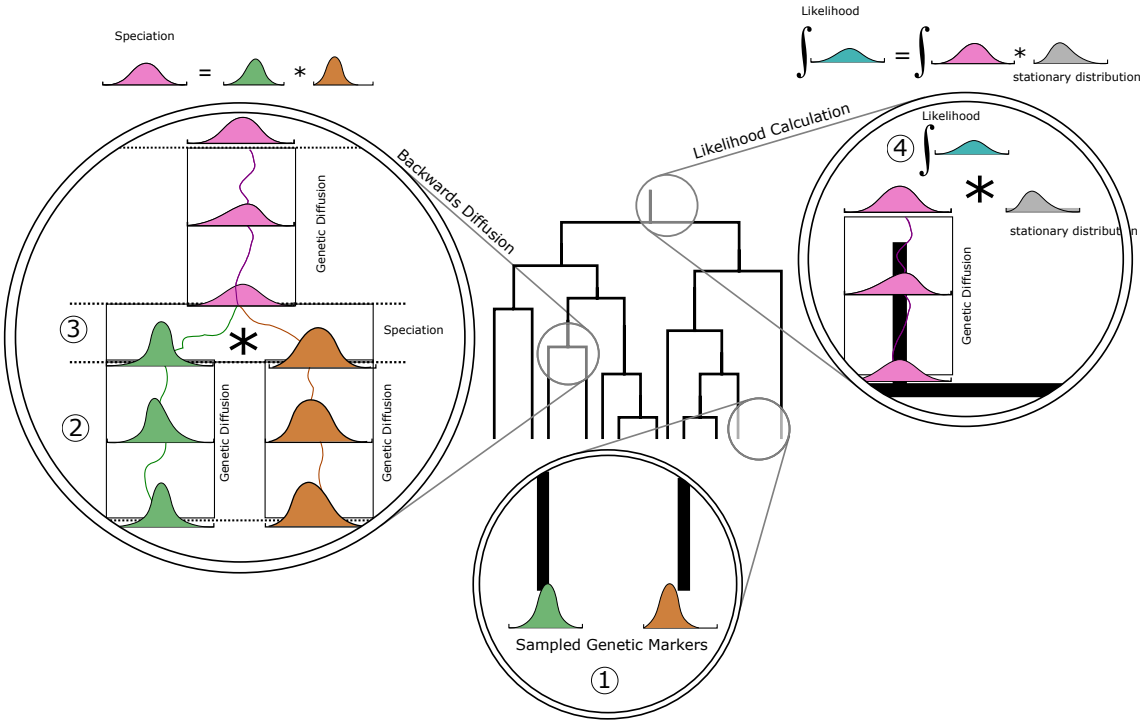


Figure 3.5: How to compute the likelihood for a site (L_m).

We finally compile all the numerical work described in this chapter into Algorithm 9. This algorithm takes $\mathcal{O}(sK\log(K))$ time per site where $\mathcal{O}(ns)$ pre-processing, where s is the number of species, K the number of Chebyshev basis functions and n is the number of individuals at a site. The pre-processing step involves counting the frequencies of allele types in each population. In practice this step could be carried out once per data set, rather than once per tree evaluated. The conversion to and from coefficients to function values in the Chebyshev expansion in lines 7 and 9 each takes $O(K\log K)$ time, using the FFT algorithm mentioned above. Approximate solution of the PDE in line 10 takes $O(K)$ time. Lastly, line 13 is carried out using root integration method 2 (as shown above).

3.9 Discussion

In this chapter we described how shifted Chebyshev polynomials are used to approximate partial likelihoods along a species tree. Most importantly we applied shifted Chebyshev polynomials in the context of spectral approximations to solve the backward diffusion (3.45) along a branch. This was an important component for efficient likelihood computation. Spectral methods are well known to be very efficient and highly accurate for approximating PDE solutions. In practise complicated boundary condi-

Algorithm 9 SNAPPER

procedure LOG-LIKELIHOOD(X, S)
  for All sites m **do**
    for All nodes i in a post-order traversal of the tree **do**
      if i is a leaf node **then**
        Evaluate $\ell_i^B(x_j)$ for each j using (3.16)
      else
        Evaluate $\ell_i^B(x_p) = \ell_j^T(x_p)\ell_k^T(x_p)$ for each p
      end if
      Compute coefficients $\lambda_{i,j}^B$ from values $\ell_i^B(x_j)$
      Compute coefficients $\lambda_{i,j}^T$ by solving the PDE (3.22) with $\boldsymbol{\lambda}(0) = \boldsymbol{\lambda}_i^B$
      Compute values $\ell_i^T(x_j)$ from coefficients $\lambda_{i,j}^T$
    end for
    Compute L_m using numerical integration.
  end for
  return $\sum_m \log(L_m)$
end procedure

Numerical algorithm implemented in SNAPPER for computing the likelihood of a species tree. Suppose that X is the data for biallelic markers and S the parameters of the tree.

tions make them difficult to implement. Luckily in the case of the backward diffusion we have shown that boundary conditions are implicitly satisfied when initial conditions are smooth. Therefore the numerical implementation was fairly straightforward without any nasty complications. We also used a very fast and stable approximation of $\exp(\boldsymbol{Q}t)$ that relies on the Caratheodory-Fejer method of matrix exponentiation. This involved solving a system of ODEs along a branch. A direct implementation requires $\mathcal{O}(K^2)$ time per branch, where K is the number of Chebyshev basis functions used in the approximation. However we showed that we can solve an equivalent sparse system in $\mathcal{O}(K)$ time. We also outlined a parallel algorithm that exploits the structure of the sparse system in special cases. Furthermore we derived some explicit expressions of the sparse system which can further expand capability of the diffusion model, such as varying mutation rates across sites. We also found an efficient solution to deal with the integration of likelihoods at the root of a tree. All of these accumulated results were implemented in SNAPPER.

Chapter 4

Inferring ecological niches of plants

4.1 Introduction

In the previous chapters we developed tools to efficiently infer species trees from molecular data. In this chapter we incorporate species trees into species distribution models (SDMs) and in doing so we extend the inference capabilities of these models.

The *ecological niche* of a species is broadly defined as the conjunction of environmental condition within which a species can maintain a stable population level without immigration (MacArthur, 1984). SDMs use environmental measures to infer a species range and preferred habitat. Fundamentally, the aim is to describe the underlying ecological niche of the species over space and time (Kearney and Porter, 2009). However most approaches do not quantify the niche but rather correlate the spatial environmental measures to species distribution records (Meier *et al.*, 2010; Mod *et al.*, 2016). A mechanistic approach to quantifying a species niche involves explicitly linking functional traits, that is traits that lead to growth and reproduction of a species, to environmental measures. A model of the interaction between functional traits provide a basis for defining a species niche which can then be projected as spatial distribution of the species.

Here we describe a mechanistic growth model for plants and define the niche of a plant species in terms of the continuous model parameters. In fact these parameters govern the growth rates of the plant biomass in terms of environmental resource availability. Using the model parameters and species trees we infer distributions of present-day plant species and their ancestral populations.

This chapter is structured as follows: In the second section we begin by introducing a framework for studying ecological niches or plants, based on a plant growth model

originally used by Higgins *et al.* (2012) to study the distribution of conifer species in Europe. In the third section we describe three niche models. The first model is the main ingredient for the two models that follow. This model quantifies an ecological niche for a single present-day species in terms of the plant growth model parameters (the continuous indices). The second model includes competition between present-day species. Environmental resources are competed for directly based on biomass ratios. The third model enables inference of ecological niches of ancestral populations. We fit a multivariate Brownian motion process on a species tree to model the evolution of ecological species parameters. At the tips of the species tree we fit a plant growth model for each species. In the fourth section we explore the three models by fitting niches for a small number of conifer species endemic to New Zealand. In the last section we discuss and outline future work.

4.2 Setting up an ecological niche for plants

The holy grail of plant ecology is understanding how physiological and morphological attributes of plants interact with environmental factors (Schimper, 1903; Tilman, 1988; Aerts, 1999). Factors such as water, heat, light and soil are all thought to play a significant role. The role that physiological and environmental factors play in shaping plant distributions was first brought to light by Schimper (1903). The Thornley Transport Resistance (TTR) model, as implemented in Higgins *et al.* (2012), is a model very much in the spirit of Schimper (1903). It models the uptake of carbon and nitrogen substances from sources (surrounding environment) to sinks (biomass of plant). See Dormann *et al.* (2012) and Thornley (1995) for variations on the model.

The TTR model in Higgins *et al.* (2012) describes plant growth via the uptake of carbon and nitrogen using a system of ordinary differential equations (ODEs). Environmental measurements are incorporated in the system of ODEs. Here we state the nonlinear ODEs without spending too much time on discussing the role of specific terms. See Higgins *et al.* (2012) for a detailed treatment. The system of ODEs is given

by

$$\frac{\partial M_s}{\partial t} = k_g(t)\frac{C_s N_s}{M_s} - k_l(t)\frac{M_s}{1+\frac{\zeta}{M_s}}, \tag{4.1}$$

$$\frac{\partial C_s}{\partial t} = \frac{M_s}{(1+\frac{M_s}{\eta})(1+\frac{C_s}{M_s\xi})}a(t) - \delta k_g(t)\frac{C_s N_s}{M_s} - \frac{M_s M_r}{M_s+M_r}\left(\frac{C_s}{M_s} - \frac{C_r}{M_r}\right), \tag{4.2}$$

$$\frac{\partial N_s}{\partial t} = \frac{M_s M_r}{M_s+M_r}\left(\frac{N_r}{M_r} - \frac{N_s}{M_s}\right) - \gamma k_g(t)\frac{C_s N_s}{M_s}, \tag{4.3}$$

and

$$\frac{\partial M_r}{\partial t} = k_g(t)\frac{C_r N_r}{M_r} - k_l(t)\frac{M_r}{1+\frac{\zeta}{M_r}}, \tag{4.4}$$

$$\frac{\partial C_r}{\partial t} = \frac{M_s M_r}{M_s+M_r}\left(\frac{C_s}{M_s} - \frac{C_r}{M_r}\right) - \delta k_g(t)\frac{C_r N_r}{M_r}, \tag{4.5}$$

$$\frac{\partial N_r}{\partial t} = \frac{M_r}{(1+\frac{M_r}{\zeta})(1+\frac{N_r}{M_r\kappa})}b(t) - \gamma k_g(t)\frac{C_r N_r}{M_r} - \frac{M_s M_r}{M_s+M_r}\left(\frac{N_r}{M_r} - \frac{N_s}{M_s}\right). \tag{4.6}$$

Briefly, M_s is the biomass of the shoot; M_r is the biomass of the root. These are collectively referred to as biomass of the plant. The biomass of a plant plays an important role for defining the likelihood of growth at a particular location (a set of environmental measures).

The biomass of a plant (4.1),(4.4) changes as a function of two forcing variables, namely photosynthesis $k_g(t)$ and litter $k_l(t)$. These functions are in turn specified in terms of minimum temperature T_{min} and maximum temperature T_{max}

$$k_g(t) = g\begin{cases} \dfrac{T_{min}(t)-\beta_1}{\beta_2-\beta_1}, & \beta_1 < T_{min}(t) < \beta_2, \\ 1, & \beta_3 < x < \beta_4, \\ \dfrac{\beta_6 - T_{min}(t)}{\beta_6-\beta_5}, & \beta_5 < T_{min}(t) < \beta_6, \\ 0, & \text{otherwise}, \end{cases}, \tag{4.7}$$

$$k_l(t) = l\begin{cases} \dfrac{T_{max}(t)-\beta_7}{\beta_8-\beta_7}, & \beta_7 < T_{max}(t) < \beta_8, \\ 0, & \text{otherwise}. \end{cases} \tag{4.8}$$

Carbon in the shoot and root (C_s, C_r) and nitrogen in shoot and root (N_s, N_r) naturally also contribute to plant growth. Furthermore change in carbon (4.2), (4.5) and

change in nitrogen (4.3), (4.6) are defined in terms of two additional forcing variables

$$a(t) = \begin{cases} a^*(t), & 0 < a^*(t) < 1, \\ 1, & a^*(t) \geq 1, \\ 0, & \text{otherwise}, \end{cases} \tag{4.9}$$

$$b(t) = \begin{cases} b^*(t), & 0 < b^*(t) < 1, \\ 1, & b^*(t) \geq 1, \\ 0, & \text{otherwise}, \end{cases} \tag{4.10}$$

where

$$a^*(t) = \alpha \min\left(\frac{T_{max}(t) - \beta_9}{\beta_{10} - \beta_9}, \frac{W(t) - \beta_{11}}{\beta_{12} - \beta_{11}}, \frac{N_s(t) - \beta_{13}}{\beta_{14} - \beta_{13}}, \frac{R(t) - \beta_{15}}{\beta_{16} - \beta_{15}}\right), \tag{4.11}$$

$$b^*(t) = \beta \min\left(\frac{T(t) - \beta_{17}}{\beta_{18} - \beta_{17}}, \frac{W(t) - \beta_{19}}{\beta_{20} - \beta_{19}}, \frac{N(t) - \beta_{21}}{\beta_{21} - \beta_{22}}\right). \tag{4.12}$$

Forcing functions (4.7)-(4.10) restrict growth of a plant based on the scarcity of environmental resources. These resources are: topsoil nitrogen concentration ($N(t)$), solar radiation ($R(t)$), soil water availability ($W(t)$) and minimum, mean and maximum temperature ($T_{min}(t), T_{mean}(t), T_{max}(t)$). Resources are represented as step functions defined in terms of actual measurements. For example, suppose we have a year of monthly measurements for topsoil nitrogen, that is $N_{[t]}$ for $t = 1, \ldots, 12$ (note we use round brackets for functions and square brackets in subscript for measurements). Then we define the topsoil nitrogen function $N(t)$ as a discontinuous step function where the step heights are defined by the monthly measurements $(N_{[1]}, \ldots, N_{[12]})$. Furthermore we define the topsoil nitrogen function $N(t)$ for all $t > 0$ by repeating the measurements.

Lastly (as seen above) each resource is associated with a set of model parameters $(\beta_i, \ldots, \beta_{i+k})$. The set of associated model parameters defines the *continuous niche index* for the particular resource. We model these continuous niche indices on a species tree in the third model of the next section. In Figure 4.1 we summarise the TTR model components and interactions with a compartmental diagram.

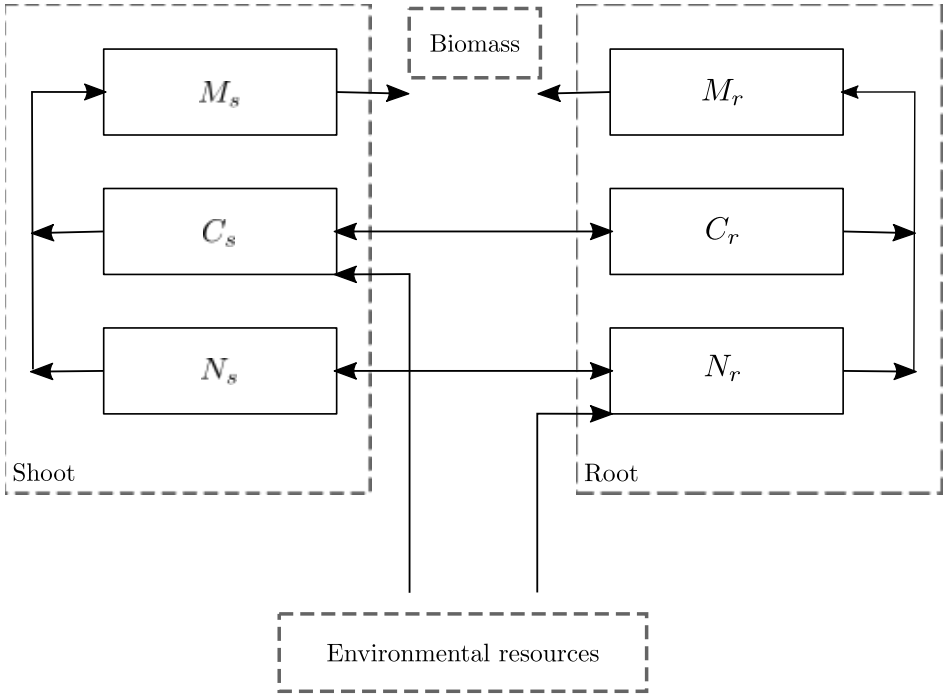


Figure 4.1: A diagram of the interacting ODE compartments of the TTR model. The compartments are divided into two process sets. Each set contains the same compartments but related to the physical structures namely the shoot and root as indicated using the subscripts. Each set consists of Biomass(M), Carbon (C) and Nitrogen (N). That is, (M_s, C_s, N_s) for the shoot and (M_r, C_r, N_r) for the root. The sets interact via transport between compartments of the same chemical element make-up and uptake of the chemicals are physiologically determined. The rate of uptake of environmental resources are determined by the mass of the shoot and the mass of the root.

4.3 Description of the models

4.3.1 A niche model for a single species

Now that we have quantified a niche in terms of the TTR model parameters we can set up a model for inferring the niche of a present day plant species from environmental data. The simplest model use the stationary solution of the system of ODEs (4.1)-(4.6). The biomass is considered *stationary* at time t^* when

$$\frac{\partial M(t)}{\partial t} = 0, \qquad \text{for } t > t^*.$$

The stationary solution for biomass is then translated into a probability of observing the plant species at a particular location. Suppose we have environmental measurements for N locations. The model likelihood is then defined in terms of probability of observing the species at each location given that we know if the species is present or absent at

the location.

In a more formal setting, suppose M_i^* is the steady state solution of the total biomass (mass of the shoot and mass of the root) of the system of ODEs (4.1)-(4.6) for location i with environmental measurements $\boldsymbol{e}_i = \{N_{[t]}, R_{[t]}, W_{[t]}, T_{min,[t]}, T_{mean,[t]}, T_{max,[t]}\}$, for $t = 1, \ldots, T$. Furthermore suppose we have TTR model parameters $\boldsymbol{\beta} = \beta_1, \ldots, \beta_{24}$. We define the probability the species will grow at location i as

$$Pr[\text{Species grow at loc } i] = 1 - \exp(-M_i^*). \tag{4.13}$$

The likelihood for a niche (TTR model parameters) given the locations $i = 1, \ldots, N$ can be defined as

$$L(a_1, \ldots, a_N|\boldsymbol{\beta}) = \prod_{\substack{i=1 \\ a_i \in \{1\}}}^{N} Pr[\text{Species grow at loc } i] \prod_{\substack{i=1 \\ a_i \in \{0\}}}^{N} (1 - Pr[\text{Species grow at loc } i]) \tag{4.14}$$

where a_i is the observation for location i (a present absent indicator, which is 1 if a species is present (observed) at location i; 0 if a species is assumed to be absent at location i).

Computing the probability of a location

In order to compute M_i^* for a site i we numerically integrate the system of ODEs (4.1)-(4.6) using a fourth-order Runge-Kutta method (Dormand and Prince, 1980; Shampine, 1986). This method is robust and is a good general candidate for numerical solution of ODEs.

4.3.2 A model for competition

We introduce a slight variation of the TTR model in order to infer joint niches of competing plant species. We assume that species compete directly for radiation and nitrogen at locations. Each species is fitted with a TTR model where the radiation and nitrogen resources are restricted based on a ratio of the biomasses for competing plant species.

More specifically, suppose we have s numbered competing species. Then we define

the ratio of the shoot mass for species i at time t

$$\alpha_s^i(t) = \frac{M_s^i(t)}{\sum_{j=1}^K M_s^j(t)} \tag{4.15}$$

where $M_s^i(t)$ is the shoot mass for species i at time t. Similarly for the ratio of the root mass for species i at time t

$$\alpha_r^i(t) = \frac{M_r^i(t)}{\sum_{j=1}^K M_r^j(t)} \tag{4.16}$$

where $M_r^i(t)$ is the root mass for species i at time t.

Now for each species i we simply substitute the equations (4.11)-(4.12) with equations that include the shoot and root ratios at the nitrogen resource function $N(t)$ and radiation resource function $R(t)$

$$a^*(t) = \alpha \min\left\{\frac{T_{max}(t) - \beta_9}{\beta_{10} - \beta_9}, \frac{W(t) - \beta_{11}}{\beta_{12} - \beta_{11}}, \frac{N_s(t) - \beta_{13}}{\beta_{14} - \beta_{13}}, \frac{\alpha_s^i(t)R(t) - \beta_{15}}{\beta_{16} - \beta_{15}}\right\} \tag{4.17}$$

$$b^*(t) = \beta \min\left\{\frac{T(t) - \beta_{17}}{\beta_{18} - \beta_{17}}, \frac{W(t) - \beta_{19}}{\beta_{20} - \beta_{19}}, \frac{\alpha_r^i(t)N(t) - \beta_{21}}{\beta_{21} - \beta_{22}}\right\}. \tag{4.18}$$

The above two equations link the formerly separate ODE systems for each species such that we have to simultaneously solve the resulting coupled and much larger system of ODEs.

The likelihood for inferring joint niches is given by taking the product of the likelihood for each species

$$L(\mathbf{A}|\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s) = L(a_{1,1}, \ldots, a_{N,1}|\boldsymbol{\beta}_1) \cdots L(a_{1,s}, \ldots, a_{N,s}|\boldsymbol{\beta}_s) \tag{4.19}$$

where $\boldsymbol{A}$ is a binary matrix of size $N \times s$ where an element $a_{i,j}$ indicates the observation for species j at location i. Presence-absence indicators for each species under this model are simulated in the same way as in the single species TTR model (see previous section).

Computing the likelihood for a joint niche

In order to compute M_i^* for a site we numerically integrate the system of ODEs (4.1)-(4.6) of all species together using a fourth-order Runge-Kutta method (Dormand and Prince, 1980; Shampine, 1986). Note that likelihoods for each species cannot be computed independently since we calculate the ratios at each timestep based on root

biomass $M_r^i(t)$ and shoot biomass $M_s^i(t)$.

4.3.3 Modelling niches on species trees

We model changes in niche indices along a species tree using multivariate Brownian motion. The use of Brownian motion to model quantitative traits is discussed in depth by Felsenstein (1988, 2004). The choice of random process is simply one of mathematical convenience since the probability distributions for niche indices at the tips of a species tree are a multivariate normal distributions.

Suppose $\mathbf{B}$ is a $s \times r$ matrix that describes niche parameters at the tips of a species tree. That is,

$$\text{vec}(\mathbf{B}) = \beta_{1,1}, \ldots, \beta_{1,r}, \ldots, \beta_{s,1}, \ldots, \beta_{s,r}$$

where r is the number of niche parameters and s is the number of species. Assuming Brownian motion along a species tree we have that

$$\text{vec}(\mathbf{B}) \sim N(\text{vec}(\bar{\mathbf{B}}), \mathbf{T} \otimes \mathbf{A}) \tag{4.20}$$

where the mean $\bar{\mathbf{B}}$ is a $s \times r$ matrix defined in terms of the niche parameters at the root $\boldsymbol{\beta}_\rho$ stacked for each species,

$$\text{vec}(\bar{\mathbf{B}}) = \beta_{\rho,1}, \ldots, \beta_{\rho,r}, \quad \text{for } i = 1, \ldots, s.$$

The covariance matrix $\mathbf{T} \otimes \mathbf{A}$ is a $(sr) \times (sr)$ matrix defined in terms of the Kronecker product of $\mathbf{A}$, the matrix of covariances among niche parameters for a specific species, and $\mathbf{T}$, the matrix where the (i, j) element is the branch length shared by the paths from the root up to tips for species i and species j, see Felsenstein (2004).

The likelihood of species niches with or without competition on a fixed tree is defined in terms of the product of partial likelihoods at the tips and the multivariate normal distribution (4.20)

$$L(\mathbf{a}_1, \ldots, \mathbf{a}_s | \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s, \boldsymbol{\beta}_\rho, \mathbf{T} \otimes \mathbf{A}) = L(\mathbf{a}_1 | \boldsymbol{\beta}_1) \cdots L(\mathbf{a}_s | \boldsymbol{\beta}_s) f(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s, \boldsymbol{\beta}_\rho, \mathbf{T} \otimes \mathbf{A}). \tag{4.21}$$

Here $\mathbf{a}_j$ is a binary vectors of size N_j (observations for species j for all locations N_j) with $j = 1, \ldots, s$. Also $f(\cdot)$ is the density of the of the multivariate normal distribution

(4.20).

To simulate data under this model we draw a sample from the multivariate normal distribution $f(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s, \boldsymbol{\beta}_\rho, \mathbf{T} \otimes \mathbf{A})$. This gives us our parameters at the leaves $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s$. Then we compute presence-absence observations for each location $i = 1, \ldots, N$ for each species $j = 1, \ldots, s$ similarly to the previous two models.

4.4 Model exploration and assessment

4.4.1 Objective

The model fitting in this section should be viewed as a proof of concept for niche models under the mechanistic plant growth model (described above). It is with this in mind that we opt to work with small datasets that are easy to handle. Small datasets also allow us to assess model behaviour and look at specific model components more easily. There is a lot of groundwork still needed to be done.

4.4.2 Conifer data

In the model exploration we considered three conifer species endemic to New Zealand namely - *Alpinus*, *Toatoa* and *Trichomanoides* of the conifer clade *Phyllocladus*. The conifer dataset used in this section forms part of a much larger dataset used in Leslie *et al.* (2012) in a study looking at conifer distributions on a global scale. Each species dataset contains approximately 50-100 observations (locations where they have been sighted around New Zealand). We also generate the same number of locations where they have not been sighted by following selection procedures in Higgins *et al.* (2012) on a square kilometre grid of New Zealand (roughly 270,000 locations). For each location we have 12 monthly measurements for each of the following environmental resources: topsoil nitrogen concentration ($N(t)$), solar radiation ($R(t)$), soil water availability ($W(t)$) and minimum, mean and maximum temperature ($T_{min}(t), T_{mean}(t), T_{max}(t)$).

4.4.3 Model fitting protocol

In order to assess the robustness of niche modelling we fit three different models in a sequence of increasing complexity: (i) A niche model for a single species. (ii) A model with competition between two species. (iii) A niche model on a three taxa species tree.

In the first experiment we check the TTR model integrity and verify convergence of

the MCMC sampler for model parameters. We used a general purpose MCMC sampler (Christen *et al.*, 2010) to conduct a Bayesian inference of TTR model parameters for the conifer species *P. Alpinus*. We have no speciation hypothesis related to environmental factors for the species. Therefore we specify uninformative uniform priors for the TTR model parameters on simplexes such that the functions (4.7)-(4.10) always hold (for example $0 < \beta_1 < \beta_2 < 1$). We initialised multiple MCMC chains with initial conditions drawn from the uniform prior. We ran each chain for 300,000 iterations. Stationarity and convergence of the MCMC chains were assessed by looking at trace plots of parameters as well as computing the number of effective independent samples. This was carried out using the software Tracer (Rambaut *et al.*, 2018). Thereafter we used the mean of the posterior distribution to assess the fitted model by computing the ODE components $(M_s, \ldots, N_r)$ for 600 timesteps (months) for all observations. Furthermore, model prediction was assessed by computing growth probabilities for all $i = 1, \ldots, 270,000$ locations across New Zealand.

For the second experiment we follow similar steps to assess MCMC convergence and stationarity. We model competition between *P. Toatoa* and *P. Trichomanoides* since they have overlapping niches. In this experiment we change the random selection process of locations where species have not been sighted. Instead of a random procedure we use observations of one species to indicate absence of the other species (since we are modelling competition). We use the mean of the posterior distribution to plot growth predictions. Here we encountered some issues with model fitting which we discuss in the next section.

For the last experiment we model niches on a three taxa tree. We used all the three species *P. Alpinus*, *P. Toatoa* and *P. Trichomanoides* and fixed the species tree to the one estimated by Leslie *et al.* (2012). The three species arose from a common ancestral population roughly 15-20 million years ago, with *P. Alpinus* and *P. Toatoa* being closer related with a common ancestor roughly $10 - 12$ million years ago. Note that fixing the species tree implies we fix the parameter T.

We follow similar steps to assess MCMC convergence as outlined above with an added uninformative Inverse-Wishart prior on the covariance matrices A with degrees of freedom equal to the number of parameters r and scale matrix set to a $r \times r$ identity matrix (we give formulas of prior densities in Appendix D) We use the same uniform prior as in the single species case for each species TTR model at the tips $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_s$ including the TTR parameters at the root $\boldsymbol{\beta}_\rho$. Next, to assess the model growth predictions we used the mean of the posteriors and computed growth probabilities

across the 270,000 locations in New Zealand for each species including the ancestral population.

For all MCMC chain simulations we used a server with Intel i3-7100 CPU.

4.4.4 Results

TTR model for a single species

It took a total of ~ 5 hours for the chain to complete with a mean effective samples size (ESS) of roughly 650. We provide a complete list of summary statistics for all fitted models in Appendix E. There were no issues with parameter convergence. In Figure 4.2 we see that the model predicts substantial biomass after the solution reach stationarity for most present sites in the dataset. We also see in Figure 4.2 that sufficient increase in Carbon and Nitrogen in both the shoot and the root lead to substantial biomass increases. In Figure 4.3 we see that the model predicts close to zero biomass after the solution reached stationarity for most absent sites in the dataset. We also see in both Figure 4.2 and Figuree 4.3 that there are no irregular or unexpected behaviour for the solutions of the TTR model components given the observations (present and absent) in the dataset. This indicates that the model is quite robust given a large variation in observations. Furthermore we project the mean of the posterior niche of *P. Alpinus* onto a square kilometre grid of New Zealand (see Figure 4.4). The model captures most of the alpine regions of New Zealand. However one striking feature of the fitted model is that there is only a slight continuous progression from high probability (green) locations to low probability (grey) locations. This might be an inherent artefact of the model (or just species specific).

Competition between two species

It took approximately ~ 11 hours to simulate a MCMC chain for 500,000 iterations with a mean ESS of roughly 600. There were no issues with convergence however we encountered some issues with the fitted model. To assess the fitted model we plotted the growth probabilities (as predicted by the mean of the posterior) of both *P. Toatoa* and *P. Trichomanoides* in Figure 4.5. We see that for *P. Toatoa* all the observations regardless of whether it is a present or absent observation have low probability of growth. On the contrary we see that for the conifer species Trichomanoides all the observation have high probability of growth. Clearly the model with competition does not describe the data well. Some work is needed to identify the issue.

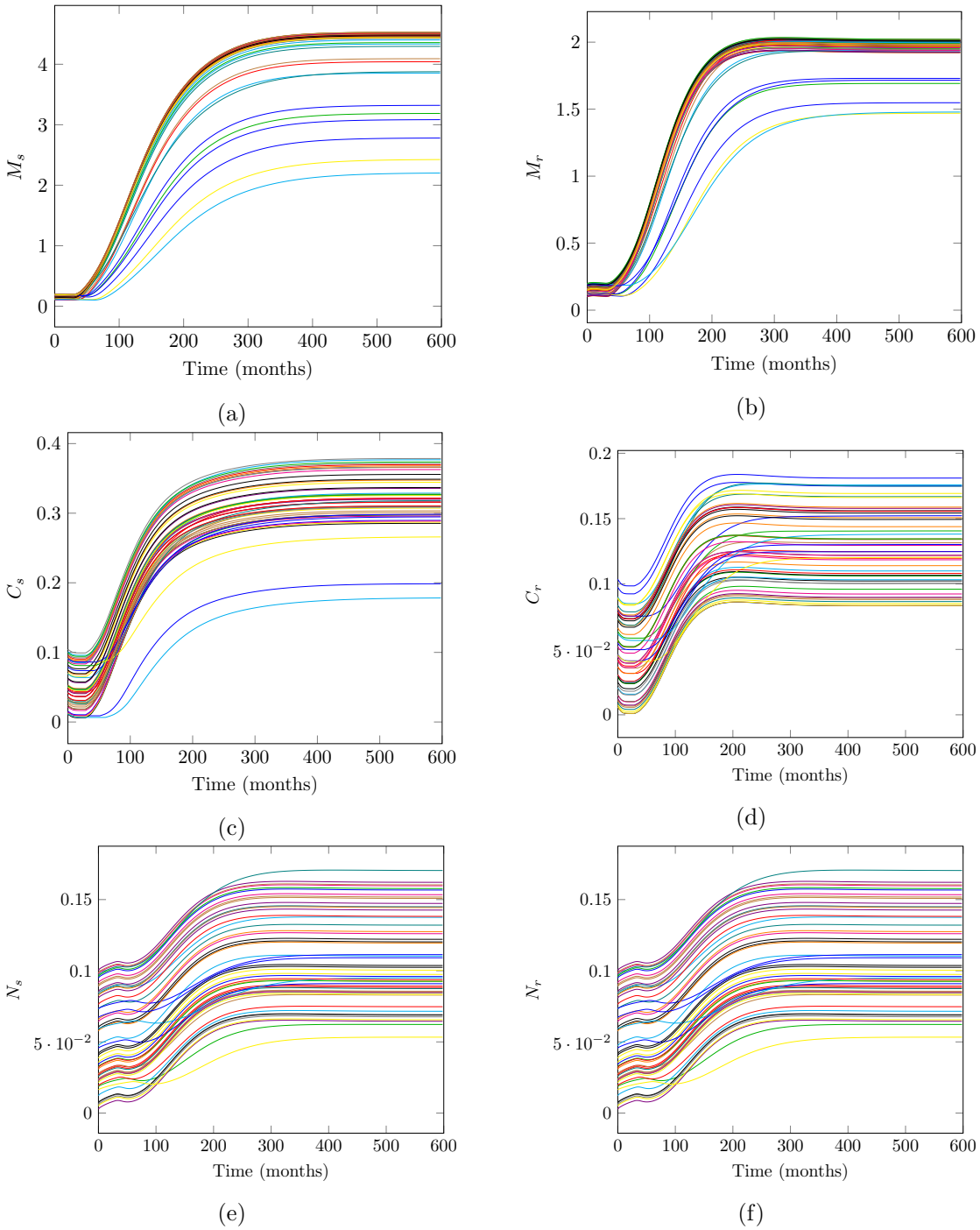


Figure 4.2: Model 1: We plot the timestep solutions up to $t = 600$ (we assume that the solution is stationary) for the ODE components $(M_s, M_r, C_s, C_r, N_s, N_r)$ of the TTR model. Each curve represents a solution to the appropriate ODE component for a present observation in the dataset of the species *P. Alpinus*. The approximate solution for a observation was computed using the mean of the posterior distribution.

Niches on a species tree

It took approximately $\sim$ 30 hours to simulate a chain for a 1,000,000 iterations with a mean ESS of roughly a 1000. In Figure 4.6(i),(ii),(iii) we see that the model captures

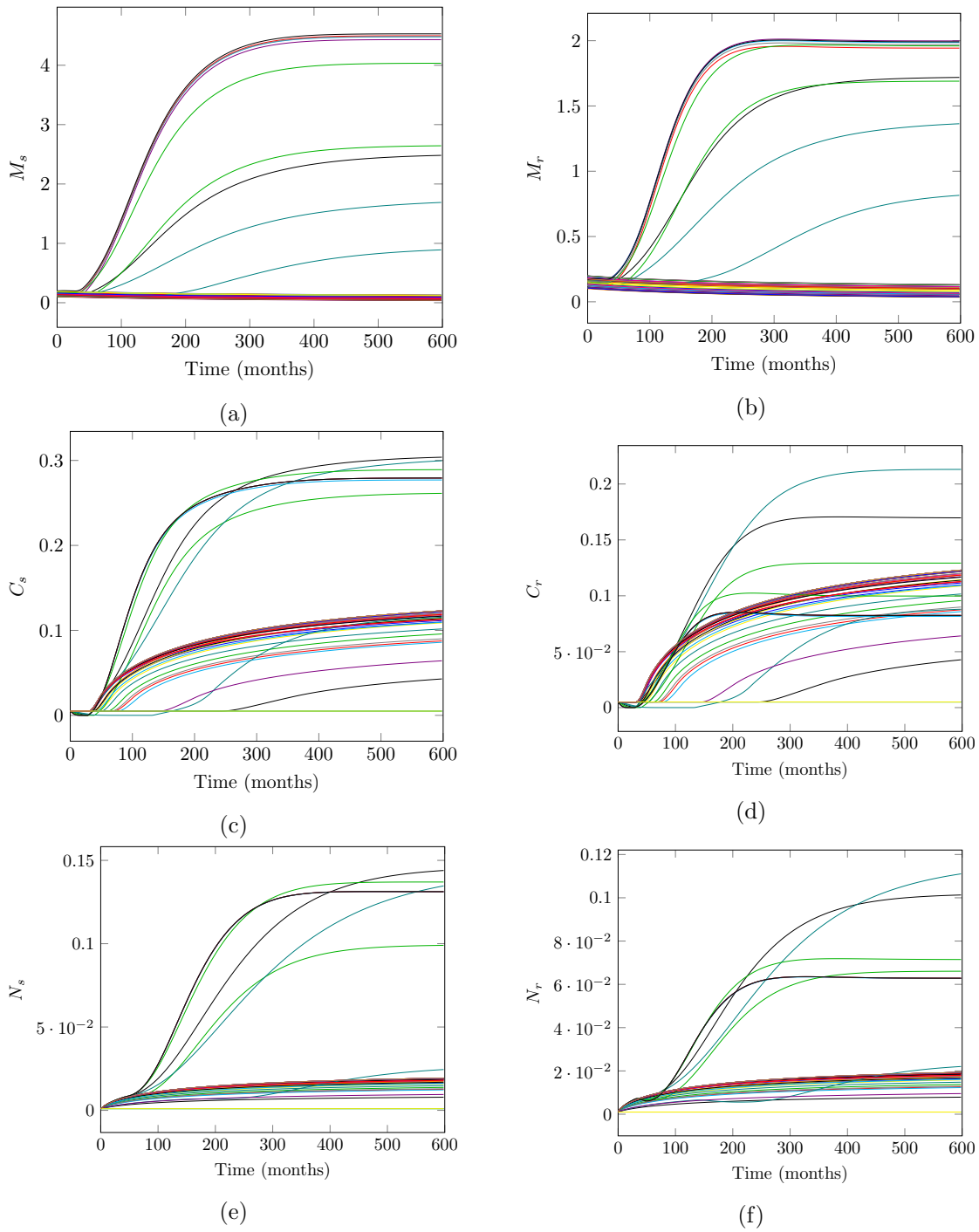


Figure 4.3: Model 1: We plot the timestep solutions up to $t = 600$ (we assume that the solution is stationary) for the ODE components $(M_s, M_r, C_s, C_r, N_s, N_r)$ of the TTR model. Each curve represents a solution to the appropriate ODE component for an absent site in the dataset of the species *P. Alpinus*. The approximate solution for a observation was computed using the mean of the posterior distribution.

some plausible niches realisation of the present day species. As expected the species *P. Alpinus* is a conifer species that is predicted to occur in alpine regions and similar

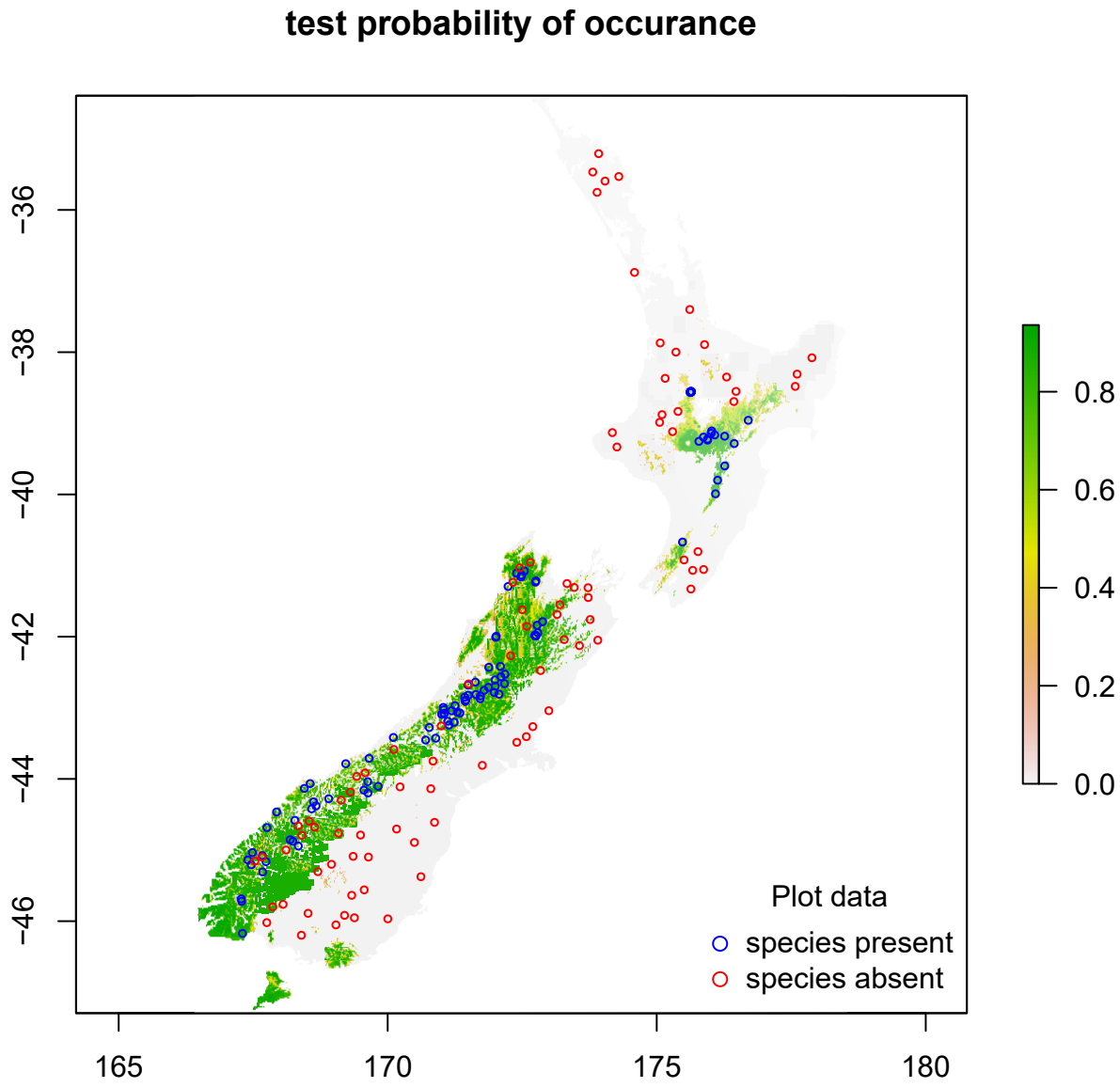


Figure 4.4: Model 1: We project the mean of the posterior niche of *P. Alpinus* onto a square kilometre grid of New Zealand. The blue dots are fixed present observation. The red dots are randomly sampled absent observations. Furthermore observation probabilities are plot on a colour grid given on the right hand side.

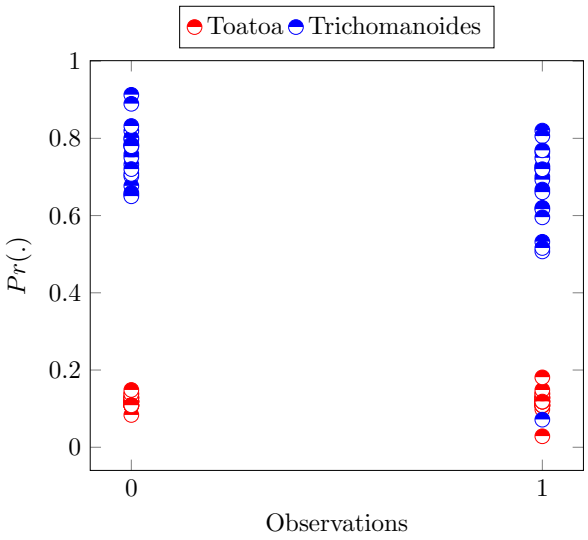


Figure 4.5: Model 2: We plot the observation probabilities for the data points used to fit the two species model with competition. Observation probabilities for conifer species *P. Toatoa* and *P. Trichomanoides* are grouped together into present observations and absent observation. Probabilities of *P. Toatoa* are red and probabilities of Trichomanoides are blue. The y-axis indicate probability of growth; the x-axis indicate whether a data point was an absent observation (0) or a present observation (1).

environments across New Zealand. The inferred niche roughly corresponds with the one in the first model fitted (see Figure 4.4). Furthermore *P. Trichomanoides* are predicted to occur in large areas of the South island but is mostly observed in tight clusters on the North island. The same is true for *P. Toatoa* which according to model prefers a similar range than *P. Trichomanoides*. The mismatch between predicted niche and observed niche could be due to the fact that a lot less present observations was used for model fitting of the latter two species. Alternatively, it could be due to competition (not modelled for this case) or that the model does not take into account whether a species had the opportunity to grow at a site. The assumption here is that it had opportunity to grow at every location. Therefore possibly indicating a large potential niche but small realised niche.

The most surprising result here is that the model predicts a viable niche for the 15-20 million year ancestor (see Figure 4.6(iv)). It is however a very small number of viable locations around the tips of the North and South islands. Nonetheless it indicates that our inferred ancestral model parameters are within a viable set of parameters (at least some of the time).

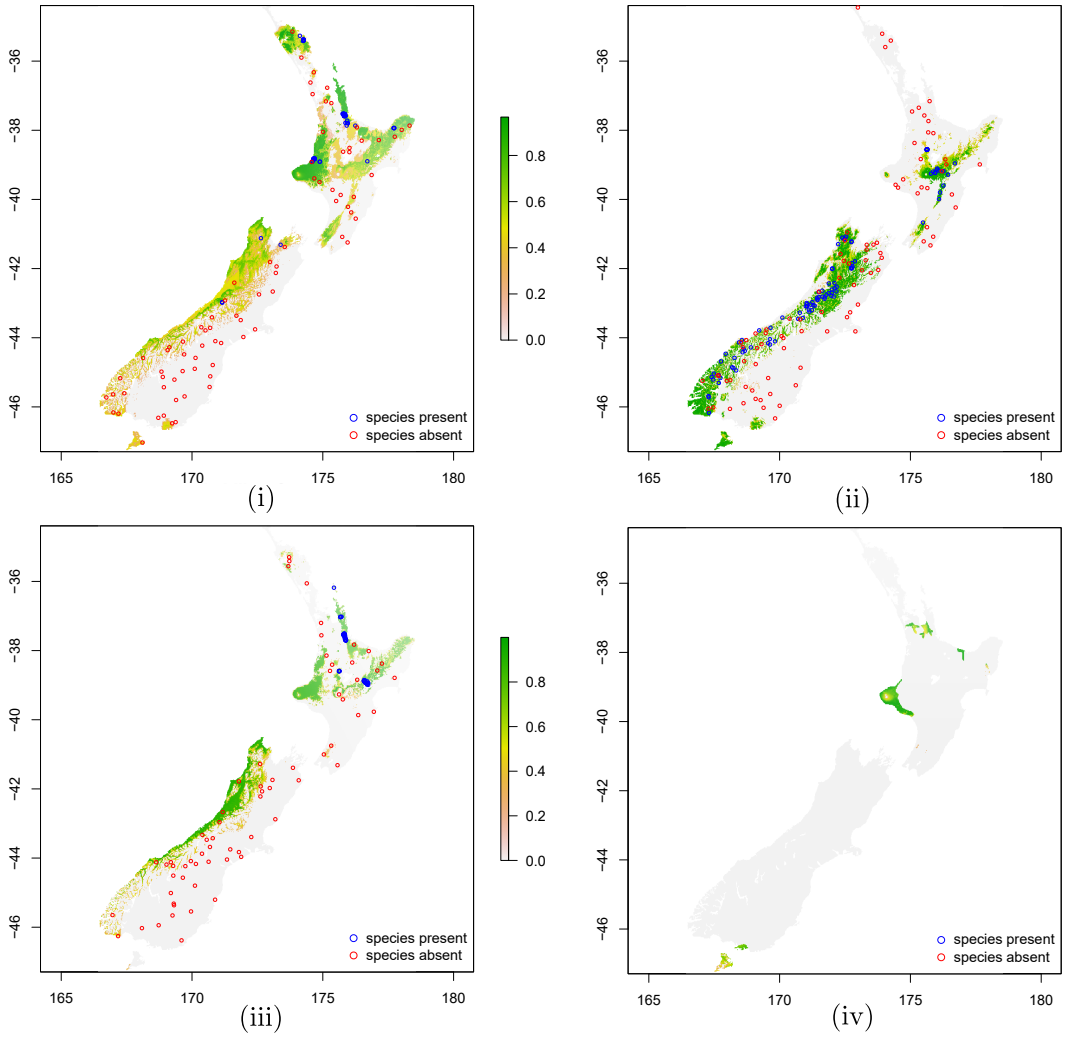


Figure 4.6: Model 3: Here we project the niches (i.e growth probabilities as estimated from the posterior distribution) of *Phyllocladus* conifers onto a map of New Zealand (i) *P. Trichomanoides* (ii)*P. Alpinus* (iii)*P. Toatoa* (iv) 15-20 million year ancestor. The blue dots are fixed present observation. The red dots are randomly sampled absent observations. Furthermore observation probabilities are plot on a colour grid given on the right hand side.

4.5 Discussion

In this chapter we present a proof of concept for a method to infer ancestral and present-day niches for plants based on a mechanistic growth model. We model the niche indices on a species tree using a simple Brownian motion.

We used a general MCMC sampler to infer model parameters. Care was still taken when selecting priors and checking convergence. We would suggest the same for anyone implementing general samplers. We reported some of the analysis performed. Mostly we focused on inference results from a conifer dataset.

The results we have presented in this chapter are interesting and potentially exciting especially the viable niche predicted for the ancestral species of conifers from the *Phyllocladus* clade. However some more work remains in order to establish the model such as

- Improving computational time of likelihood evaluations.
- Checking how informative the likelihood is (we suspect we have very little information on the covariance matrix $\boldsymbol{A}$).
- Setting up experiments to check recoverability of parameters by MCMC sampler.
- Including uncertainty around the species tree.
- Verifying model predictions of ancestral niches.

We give more detail below.

4.6 Future work

First and foremost, in order to conduct experiments effectively we need to improve the computational time of likelihood calculations. For initial model exploration we used a very robust ODE integrator. However likelihood evaluations will become very slow if model fitting is scaled up to infer parameters for large number of species. One possible approach to improve computational time of likelihood evaluations is to use simpler but faster forward finite differencing methods (Iserles, 2009). Some careful work will be needed to quantify approximation error. Another approach would be to investigate simplifications of the TTR model (that is reducing the number of TTR model components). Both of these approaches can be incorporated in MCMC sampling strategies such as delayed acceptance sampling (Cui *et al.*, 2011, 2019).

After we have implemented efficient sampling methods we can set up a simulation study to verify recoverability of parameters. See Chapter 3 for an example.

In order to investigate how informative the likelihood is (particularly for the covariance matrix) we can compute the posterior distribution using different priors and different amounts of data. Specifically we can compare inference by varying the size of the species tree. We expect that an informative likelihood with different priors will not change the posterior distribution much given a reasonable amount of data.

During the model exploration we used a fixed species tree. Methods such as SNAPP and SNAPPER estimate a posterior distribution for the species tree. Therefore we can incorporate the uncertainty around a species tree in the model by drawing from the estimated posterior distribution of the species tree for each likelihood calculation in the niche model. Another interesting experiment would be to infer a species tree using the niche model. However we expect that present-day niches will contain very little information about the species tree. Therefore using large amounts of data will be important. A possible candidate dataset will be the 427 conifers species dataset used in Leslie *et al.* (2012). However we should keep in mind that this will significantly increase the parameters, $\mathbf{T} \otimes \mathbf{A}$ instead of just $\boldsymbol{A}$, and we suspect we already have very little information about $\boldsymbol{A}$.

The most challenging task that remains is to verify ancestral niches of species as predicted by the model. Fossil records are quite patchy and unreliable. Nonetheless it would be interesting to check the correlation between model predictions and existing hypotheses of speciation (that may or may not be based on fossil records). A possible candidate group would be Angiosperms. They have a reasonable fossil record that points at their ancestral range. Finally, we have outlined a niche model in an evolutionary context that can help formulate speciation hypotheses.

Chapter 5

Model fitting using graphical processing units

5.1 Introduction

Specialised computational hardware can optimise data throughput to improve runtime of model fitting procedures. Specifically, likelihood computations for models in phylogenetics using SNP datasets (assuming SNPs are independent) can be performed in parallel on Graphical Processing Units (GPUs). In fact most model fitting procedures where datapoints are assumed to be independent can be optimised on GPUs. In this chapter we design and implement a model fitting algorithm for GPUs using a concrete example (a Hidden Markov model with application in seismology). The aim is to get a better understanding of hardware capabilities and hardware limitations in order to design more elaborate algorithms for models in molecular phylogenetics.

Hidden Markov models (HMMs) are general purpose models for time-series data widely used across the sciences because of their flexibility and elegance. The model consists of one random process which can be observed with a second underlying (and unobserved) random process driving the randomness of the observed process. Therefore fitting hidden Markov models means finding the parameters of the unobserved process given time series data for the observed process. See Zucchini *et al.* (2017) for a general introduction.

Fitting HMMs can often be computationally demanding and time consuming, particularly when the number of hidden states is large or the Markov chain itself is long. In this chapter we introduce a new graphical processing unit (GPU) based algorithm designed to fit long chain HMMs, applying our approach to an HMM for nonvolcanic

tremor events developed by Wang *et al.* (2018). Even on a modest GPU, our implementation resulted in a 1000-fold increase in speed over the standard single processor algorithm, allowing a full Bayesian inference of uncertainty related to model parameters. Similar improvements would be expected for HMM models given large number of observations and moderate state spaces (< 80 states with current hardware).

Recent evidence suggest that nonvolcanic tremors are observed in close association with slow slip events, however the causal relationship between the two phenomena is not yet well understood. Slow slip events (SSEs), a type of slow earthquakes, play an important role in releasing strain energy in subduction zones, the region where one tectonic plate moves underneath another tectonic plate and sinks. It is currently understood that SSEs occur as shear slips on the bottom tip of subduction zones that transition between a fixed region above and slipping region below (Beroza and Ide, 2011). Classifying nonvolcanic tremors helps to better understand SSEs but can be time consuming when typically done by hand.

Recently, an automated procedure was developed by Wang *et al.* (2018) to classsify spatio-temporal migration patterns of nonvolcanic tremors. The procedure classifies tremor source regions into distinct geographical clusters using a Hidden Markov Model. The model is fitted using the Expectation Maximisation (EM) algorithm. Here we implement a Bayesian approach. However, fitting the model in either a frequentist framework or Bayesian framework is extremely demanding computationally, often taking days or weeks for a large dataset with moderate state space. Fortunately, technological advances in hardware have the potential to solve this issue. Specifically, we make use of fast and affordable GPUs.

In recent years HMM algorithms on GPUs have been implemented in various fields. A non-exhaustive list includes implementations in bioinformatics (Yao *et al.*, 2010), speech recognition (Yu *et al.*, 2015), a registered patent in speech matching (Chong *et al.*, 2014) and workload classification (Cuzzocrea *et al.*, 2016), as well as HMMer (Horn *et al.*, 2005) an open-source project for use with protein databases. The HMM implementations are application specific often with large number of states and mostly focused on increasing throughput of the Verterbi and Baum-Welch algorithms (Zhang *et al.*, 2009; Li *et al.*, 2009; Liu, 2009). This leads to a range of concurrent approaches. Here we focus on the efficient implementation of the forward algorithm of an HMM model given a large number of observations and a moderate number of states.

The outline of the chapter is as follows: In Section 5.2 we describe the HMM for

classifying nonvolcanic tremors. In Section 5.3 we discuss the computational framework for the GPU algorithm and also look at the hardware architecture. In Section 5.4 we discuss performance of the OpenCL implementation and compare it to the standard Forward algorithm. In Section 5.5 we report our analysis on a large tremor dataset from the Shikoku region, Japan. Lastly we discuss some future work for models in phylogenetics.

5.2 Hidden Markov model for classifying non-volcanic tremors

Non-volcanic tremor activity is clustered spatially and each spatial cluster seems to recur episodically. To represent this phenomenon using an HMM, Wang *et al.* (2018) introduce one hidden state for each spatial cluster. The tremors themselves (including the absence of a tremor) are the observations. The frequency and spatial distribution of tremors changes according to the hidden state.

More formally, we suppose that the observations of nonvolcanic tremors are a sample path of a stochastic process

$$X_t \qquad \text{for } t = 0, \ldots, N$$

with observations represented in the state space

$$I = \{\emptyset, \mathbb{R}^2\}$$

generated under an HMM with K numbered hidden states. For each hidden state $k = 1, \ldots, K$ we introduce parameters p_k, $\boldsymbol{\mu}^{(k)}$ and $\boldsymbol{\Sigma}^{(k)}$, where p_k is the probability of observing a tremor and $\boldsymbol{\mu}^{(k)}$, $\boldsymbol{\Sigma}^{(k)}$ are the mean and variance of a bivariate normal distribution modelling where a tremor is likely to occur, if it does occur.

To simplify notation we introduce for each observation x a $K \times K$ diagonal matrix $\mathbf{P}(x)$, also called the *emission matrix*, with the kth diagonal element corresponding to the probability of observing x given state k

$$\boldsymbol{P}(x)_{kk} = \begin{cases} p_k \phi(x | \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}), & \text{if a tremor occurred} \\ 1 - p_k, & \text{otherwise.} \end{cases} \tag{5.1}$$

Algorithm 10 Simulate time-series tremor observations

procedure TREMORSIM($\{p_k, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}\}_{k=,1\ldots,K}, \boldsymbol{\Gamma}$)
    Draw hidden state k from stationary distribution $\boldsymbol{\delta}$ of $\boldsymbol{\Gamma}$
    Draw from normal density $f(\boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)})$ with probability p_k
    for each timestep $t = 1, \ldots, N$ **do**
        Simulate hidden state k using transition matrix $\boldsymbol{\Gamma}$
        Draw from normal density $f(\boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)})$ with probability p_k
    end for
end procedure

Simulating time-series data under a Hidden Markov model with hidden states $k = 1, \ldots, K$ and parameters $p_k, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}$ where $\boldsymbol{\Gamma}$ is transition matrix between hidden states.

Here $\phi(.)$ is the density function of bivariate normal distribution. Let $\boldsymbol{\Gamma} = (\Gamma_{ij})$ denote the $K \times K$ transition matrix of the HMM, where Γ_{ij} indicate the transition probability from hidden state $K = i$ to $K = j$. Also, let $\boldsymbol{\delta} = \delta_1, \ldots, \delta_K$ denote the vector of probabilities for the initial state.

Suppose we have hidden states $k = 1, \ldots, K$ with parameters $p_k, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}$ and transition matrix $\boldsymbol{\Gamma}$. We simulate tremor activity as a random process X_i under the model as follow:

We start by drawing a hidden state from the stationary distribution $\boldsymbol{\delta}$ of transition matrix $\boldsymbol{\Gamma}$. Thereafter for each time step we determine the hidden state k associated with X_t from the previous step using the transition matrix $\boldsymbol{\Gamma}$. Now to generate X_t we first use p_k to determine whether a tremor event was observed. If so, we draw the geographical location of the tremor from the bivariate normal density $f(\mu_k, \Sigma_k)$. We repeat this process for $t = 1, \ldots, N$. We give more detail on simulating time-series data under the model in Algorithm 10.

Now the likelihood function for the parameters given the observed data can be written as

$$\mathrm{L}\big(\boldsymbol{\Gamma}, \boldsymbol{\delta}, \{p_k, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}\}_{k=1,\ldots,K} | \mathbf{x}_0, \ldots, \mathbf{x}_N\big) = \boldsymbol{\delta}^T \boldsymbol{\Gamma} \mathbf{P}(\mathbf{x}_0) \ldots \boldsymbol{\Gamma} \mathbf{P}(\mathbf{x}_N) \mathbf{1}. \tag{5.2}$$

.

5.3 GPU computing framework

GPUs have had a large impact across statistical and computing sciences due to cost-effective parallelism (Kindratenko, 2014). However in order to translate an algorithm from CPU to GPU some careful consideration is needed in terms of:

1. Reducing latency (how to concurrently execute instructions on GPU in order to optimise data throughput.)

2. Managing memory (how to effectively distribute and utilise memory across processors to avoid bandwidth bottlenecks).

3. Designing robust algorithms with respect to varying GPU architecture between models and vendors as well as the rapidly changing landscape of computational hardware.

Frameworks like OpenCL and CUDA, allow programmers to implement GPU algorithms with some level of generality. The implementation we describe here was carried out in the OpenCL framework. OpenCL is an open standard maintained by the non-profit technology consortium Khronos Group, see `https://www.khronos.org` for more details on the non-profit organisation.

The OpenCL framework consists of a *host* (CPU; terms in brackets relate to computation on GPU architecture) controlling one or more *compute devices* (We just used one GPU). Each compute device (GPU) is divided into *compute units* (streaming multiprocessors). Compute units are further divided into *compute elements* (microprocessors or cores). Each compute unit has access to global memory of the compute device. This access though is slow. Each compute unit also has a shared memory to allow efficient data exchange between compute elements. Each compute element has exclusive access to private memory (registers) for computation.

Note however that layout and type of compute elements change depending on the type of computational demand of the market. Tensor cores (cores that can do vector multiplication in one clock cycle) is one such an example since deep neural nets, a current hot topic, does a lot of vector computations.

In Figure 5.1 we give an example of typical architecture of current streaming processors.

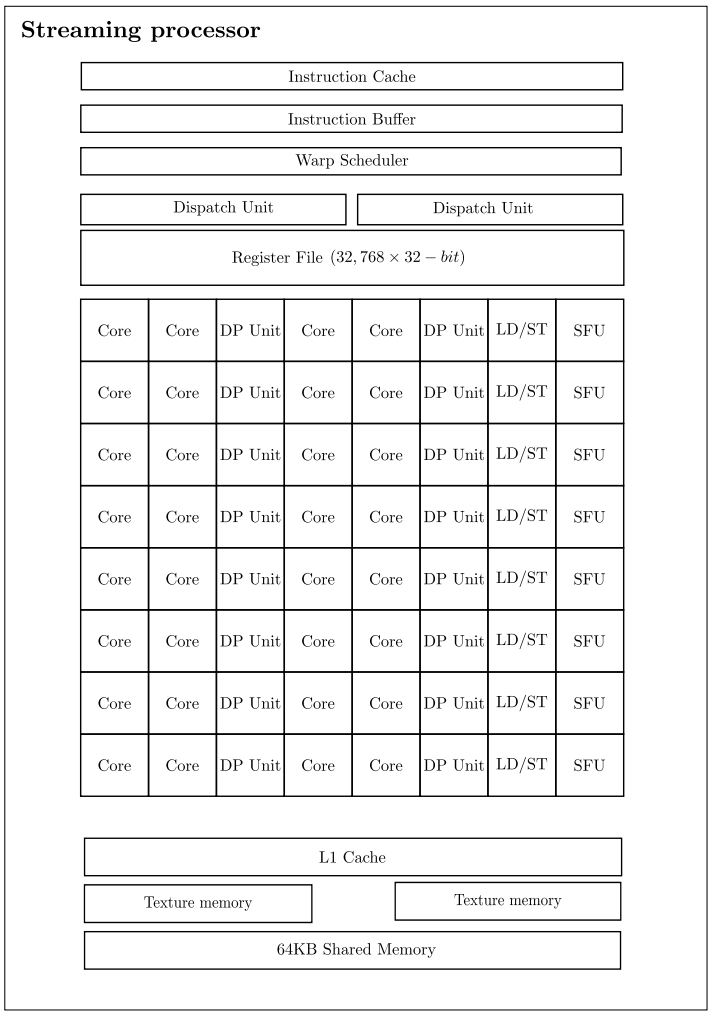


Figure 5.1: Current architecture of a typical streaming processor. It is the job of the warp scheduler to arrange computing tasks (stored in the Instruction cache and buffer) into sets of 32, called warps. Dispatch units distribute warps across compute elements to be executed in parallel. The different type of compute elements in this example are: 32-bit compute elements (cores). Cores do most of the heavy-lifting however other hardware operations are also sometimes required. 64-bit compute elements (DPU) are used when high accuracy is required for an instruction. The Load/store compute elements (LS/ST) calculate source and destination addresses for inputs and outputs. Special function compute elements (SFU) calculate transcendental functions such as $\sin, \cos, \sqrt{}$, etc. The different types of on-board memory are as follow: Register files are private memory for compute elements. Each core for instance can store 1024 32-bit elements. L1 cache is just read-only memory for fast access. Texture memory is read only memory with additional filtering that performs floating point interpolation as part of the reading process. Shared memory is programmable and exploited by programmers to share data between cores.

5.4 The likelihood algorithm

5.4.1 Overview

Our implementation will work well on a range of GPU models. For our studies we used a NVIDIA GeForce GTX 1080 Ti GPU with 28 compute units (streaming multiprocessors) each with 48KB of shared memory, 128 compute elements (cores) and a register file that can contain up to 32,768 32-bit elements distributed across the compute elements (cores). For the host we used an Intel Core i7-7700K CPU at 4.20GHz. There are two main limitation for the OpenCL algorithm in terms of hardware specifications

1. The number of registers per compute element.

2. The size of shared memory on a compute unit.

For example given the hardware described above we have (32,768/128)=256 registers per compute element. This implies that we can store up to roughly 200 32-bit matrix elements on a compute element (we also need some registers left to store counters and other meta variables). Our implementation assumes that at least two matrix rows can fit into the registers of a compute element. This gives an upper limit for the number of hidden states of $K < 100$. In order to efficiently distribute rows of a matrix and update matrix elements we need space for two matrices in the shared memory of the compute unit. Our configuration has 48KB of shared memory per compute unit. Implying that we can fit a total of $(48 \cdot 2^{10})/4 = 12288$ 32-bit matrix elements per compute unit. This gives a second upperlimit for number of hidden states of $K < 80$. To handle a large number of states, alternative parallel computing strategies should be used (Horn *et al.*, 2005; Yu *et al.*, 2015).

First we consider how the algorithm for the likelihood (5.2) would be implemented on a single processor unit. To avoid matrix-matrix multiplications we would start with the stationary vector $\boldsymbol{\delta}$ on the left, and then sequentially multiply that by transition matrices and emission matrices:

In Algorithm 11 running time will be dominated by the matrix-vector multiplication in steps 5 and 6, taking $\mathcal{O}(K^2)$ time per iteration. Hence the running time, or work, for this implementation is $\mathcal{O}(NK^2)$. Next we compare it with the parallel implementation.

The overview of our implementation is as follows:

1. We compute all of the emission matrices $\boldsymbol{P}(x_0), \ldots \boldsymbol{P}(x_N)$ in parallel.

Algorithm 11 The Forward algorithm on a CPU

```
1: procedure COMPUTE-LIKELIHOOD({p_k, μ^(k), Σ^(k)}_{k=1,...,K}, {x_0, ..., x_N})
2:     v ← δ^T
3:     for t from 0 to N do
4:         Compute P(x_t)
5:         v ← vΓ
6:         v ← vP(x_t)
7:     end for
8:     return v1
9: end procedure
```

The Forward algorithm for computing likelihood of a HMM with hidden states $k = 1, \ldots, K$ and parameters $p_k, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Sigma}^{(k)}$. Here $\boldsymbol{\Gamma}$ is the transition matrix between hidden states and $\boldsymbol{\delta}$ is its stationary distribution. Tremor observations are indicated by $\mathbf{x}_0, \ldots, \mathbf{x}_N$.

2. We then multiply the emission matrices by the transition matrices, all in parallel storing N matrices $\boldsymbol{\Gamma}\boldsymbol{P}(x_0), \ldots, \boldsymbol{\Gamma}\boldsymbol{P}(X_N)$.

3. We multiply the matrices $\boldsymbol{\Gamma}\boldsymbol{P}(x_0)), \ldots, \boldsymbol{\Gamma}\boldsymbol{P}(X_N))$*togetherinparallelinsteadofcomputing*$\boldsymbol{\Gamma}\boldsymbol{P}(x_0$ as part of a single sequence of vector-matrix multiplications.

This *increases* the work done: we are carrying out matrix-matrix multiplications instead of matrix-vector multiplications, but it allows us to spread the computation over multiple processors. We now discuss steps (1) to (3) in greater detail.

5.4.2 Step 1: Emission probability evaluation on GPU

The goal in this step is to compute the emission matrices $\mathbf{P}(x_i)$ for each observation x_i. The emission probability is defined by (5.1) and makes use of the parameters $p_k, \boldsymbol{\Sigma}_k, \boldsymbol{\mu}_k$ for each hidden state k. These parameters are initially copied to the registers of each core and remains there until all the datapoints have been evaluated. The compute elements work in parallel. Each is allocated a data point x_i, uses the stored values to compute $\boldsymbol{P}(x_i)$ and copies the diagonal matrix computed to global memory. Note that a compute element can request and copy the next data point at the same time as it processes the current data point.

For this step there is no data sharing between compute elements, allowing for data-level parallelism. Therefore it is more efficient to allow compute device compiler to optimise the work-load scheduling and data transfer between compute units in order

to fully utilise SIMD (Single instruction multiple data) instructions. Output from compute elements are collected and copied to global memory to form the list of new inputs $\{\mathbf{\Gamma}, \mathbf{P}(x_0), \ldots, \mathbf{P}(x_N)\}$ for the next kernel.

5.4.3 Step 2: Transmission-emission matrix multiplication on GPU

During the next step we compute $\mathbf{\Gamma}\boldsymbol{P}(x_i)$ for all data points x_i, again in parallel. At this point we run into limitations with memory. While the register of a single compute element is large enough to store the diagonal matrix $\boldsymbol{P}(x_i)$, it is not large enough to store the full transition matrix $\mathbf{\Gamma}$ nor the product matrix $\mathbf{\Gamma}\boldsymbol{P}(x_i)$. The solution is to break down the multiplication of $\mathbf{\Gamma}$ and $\boldsymbol{P}(x_i)$ by computing only a few rows at once.

We query the register size for each compute element to determine how many rows of $\mathbf{\Gamma}$ can be copied. The rows remain in the register until all data points have been evaluated. Thereafter the next set of rows is copied into the registers and the data points is evaluated again until all the rows of $\mathbf{\Gamma P_r}$ for $r = 0, \ldots, N$ have been computed. As $\boldsymbol{P}(x_i)$ is diagonal, the product of rows of $\mathbf{\Gamma}$ with $\boldsymbol{P}(x_i)$ is computed by simply rescaling the corresponding columns.

The next diagonal matrix subset is requested while scaling the matrix subset of the current data point. Again, there is no data sharing between compute elements, allowing for optimal data-level parallelism. Output from compute elements are collected and a new list of inputs, namely $\{(\mathbf{\Gamma P_0}), \ldots, (\mathbf{\Gamma P_N})\}$ is compiled for the final GPU kernel.

5.4.4 Algorithms as trees

Before we test our implementation of the GPU algorithm we point out the computational trees that describe some of the algorithms and procedures previously discussed (i.e forward algorithm of an HMM; transmission-emission matrix multiplication on GPU). For these particular trees a node at the bottom of a parent branch i are output from vector or matrix operations. These operations are represented by the coalescing child branches j and k. Nodes at the bottom of the child branches indicate inputs for the matrix or vector operations. Operations are performed starting at the bottom of the tree moving upwards until the root node is reached (final output). The main idea of arranging operations in a tree structure is to indicate operations that can be executed in parallel (i.e multiple coalescing branches with parent and child nodes at the same height). In Figure 5.2 we show the computational structure of the Forward algorithm

of an HMM. Serial operations are nicely represented by a caterpillar tree. Caterpillar trees do not have multiple coalescing branches at the same height. Hence there is no opportunity to exploit parallization. The algorithm takes N steps to execute.

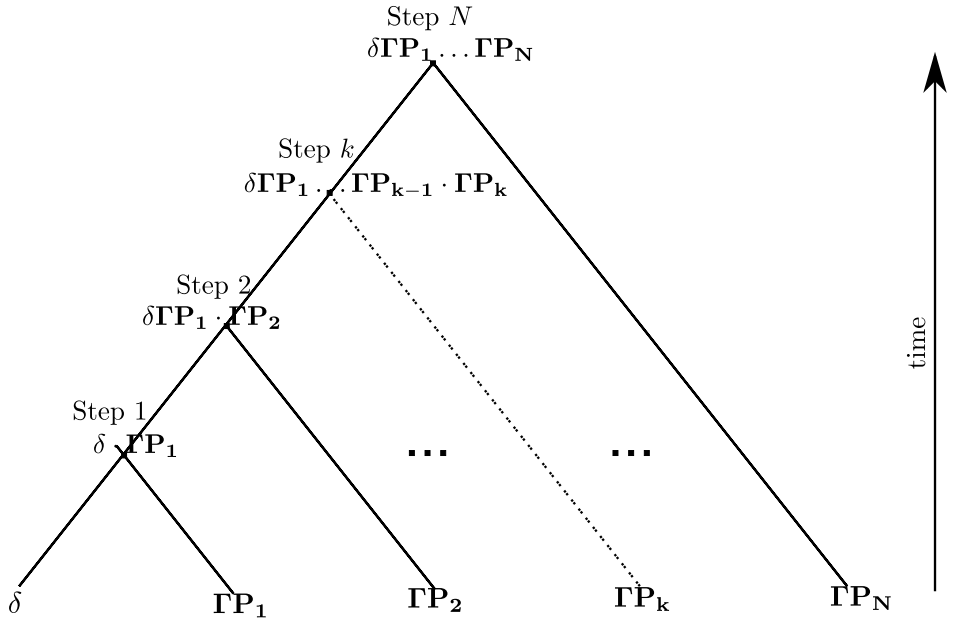


Figure 5.2: Computational tree for the Forward algorithm. Operations are executed starting at the bottom of the tree moving upward. We see that for each step there is only one pair of coalescing branches.

In Figure 5.3 we show the computational tree of a highly parallel transmission-emission matrix multiplication as a balanced tree. This transmission-emission matrix multiplication will take $N \log N$ steps to execute. In theory this is the optimal number of steps for computing a square matrix chain. We assume however that in each step we can execute at least $N/2$ of computations in parallel. In practise however, at least given current hardware, we are restricted. Therefore it is not an optimal algorithm for implementation. Nonetheless thinking in terms of computational trees is still a useful exercise when designing parallel algorithms.

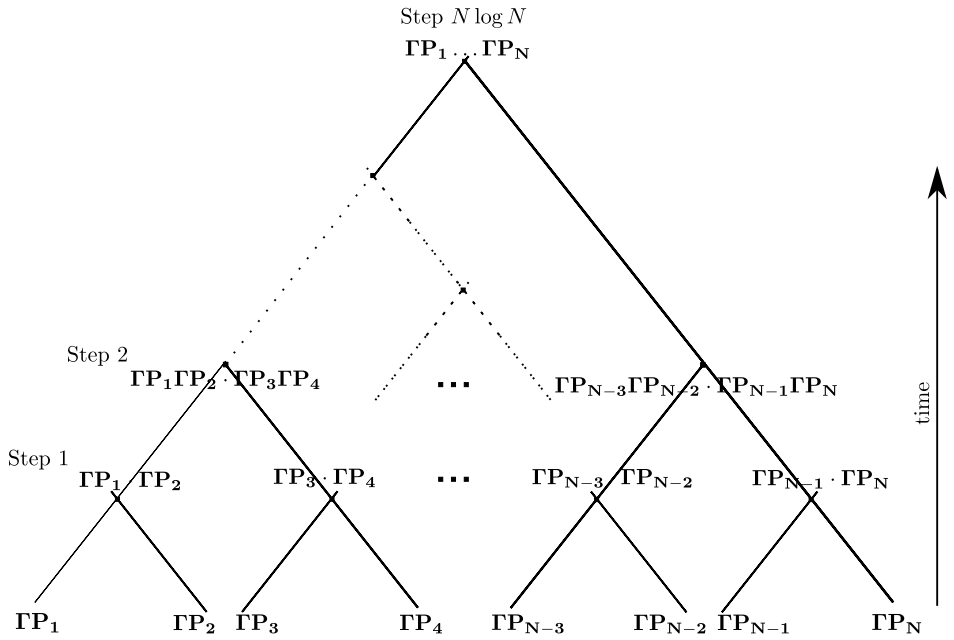


Figure 5.3: Computational tree for the transmission-emission matrix multiplication in step 2 of the GPU algorithm. We see that for each step there is multiple pairs of coalescing branches.

5.5 Performance assessment of OpenCL implementation

5.5.1 Protocol

We investigate the performance of the OpenCL implementation by conducting the following numerical experiments: (i) An error assessment storing 64-bit values compared to 32-bit values throughout the log-likelihood computation. (ii) A comparison of computational time between OpenCL algorithm and Forward algorithm on CPU. (iii) A comparison of computational time between matrix chain multiplication using OpenCL algorithm and matrix chain multiplication using popular GPU libraries.

In the first experiment, to check for possible error accumulation we executed the same algorithm on a GPU using 32-bit values and on a CPU using 64bit values and compare the relative difference. First we fixed the number of HMM states to $K = 25$ while increasing the number of datapoints over a range of magnitude orders $N = 10^2, \ldots, 10^5$. Thereafter we fixed the number of datapoints to $N = 100,000$ and increased the number of HMM states for $K = 5, 10, \ldots, 50$.

In the second experiment we compared computational times of the GPU algorithm

with the Forward algorithm from the software library Tensor flow. First we fixed the number of HMM states to $K = 25$ while increasing the number of datapoints over a range of magnitude orders $N = 10^2, \ldots, 10^5$. Thereafter we fixed the number of datapoints to $N = 100,000$ and increased the number of HMM states for $K = 5, 10, \ldots, 50$. In each case model parameters were drawn from the prior distribution (discussed in the next section) and thereafter data was simulated using the R software package in Wang *et al.* (2018).

In the third experiment we specifically compared computation time of step 3 in the OpenCL algorithm with matrix-chain multiplication using popular GPU BLAS (Basic Linear Algebra Subprograms) libraries. We used subroutines from the CLBlast library as well as the MAGMA BLAS library to do the matrix-chain multiplication. CLBLast is a general BLAS library in OpenCL that automatically tunes subroutines for specific hardware based on compile time. MAGMA BLAS is a CUDA library exclusively available for NVIDIA GPUs. We followed the same procedure as in the previous two experiments except that we fixed the number of HMM states to $K = 50$.

All experiments were done using a NVIDIA GeForce GTX 1080 Ti GPU with 28 streaming processors each with 48KB of shared memory, 128 cores and a register file that can contain up to 32,768 32-bit elements distributed across the cores. Furthermore we used an Intel Core i7-7700K CPU at 4.20GHz.

5.5.2 Results

We see that the accuracy loss due to 32-bit implementation is insignificant in both sequence length and matrix size experiments (see Figure 5.4). More importantly we see the GPU algorithm executes orders of magnitude faster then a Forward algorithm (see Figure 5.5 and Figure 5.6). As the number of data points increase the forward algorithm increases linearly (as expected) and the GPU algorithm increases polynomially (see Figure 5.5). This is due to the overhead and delay increases from transferring data back and forth between different levels of the memory hierarchy. Therefore we do not see this when increasing the matrix size (see Figure 5.6). Here both algorithm increase linearly.

In the matrix chain multiplication (see Figure 5.7 and Figure 5.8) using the MAGMA library outperforms the OpenCL algorithm for small matrices. This is due to specific optimisation for each case of matrix size. These optimisations are tedious and improvements are only marginal. Furthermore implementing these libraries in the OpenCL algorithm is not straightforward due to small tweaks and scaling coefficients that we keep

track of in addition to performing the matrix-chain multiplication. We note that the OpenCL algorithm became very slow (not shown here) when the HMM had more than 100 states. This is due to memory limitations of hardware, as previously discussed.

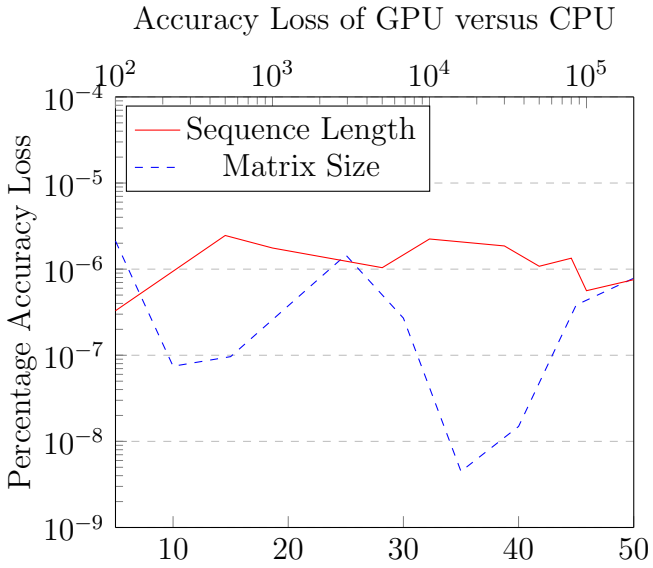


Figure 5.4: We indicate the relative difference of the two likelihood outputs on the y-axis. The red line plots relative difference as number of datapoints N increase while number of states K are fixed, top x-axis indicate the number of data points in orders of magnitude. The blue line plots relative difference as number of HMM states K increase while number of datapoints are fixed, bottom x-axis indicate the number of states.

5.6 Bayesian analysis of nonvolcanic tremor data

5.6.1 Model priors

Prior distributions are important and need to be carefully considered and justified. It is known that tremors occur in sequence bursts that cluster around the same area (Wang *et al.*, 2018). This observation we translate into the model by specifying a model prior centred around sparse transition matrices. More formally, we specify a symmetric Dirichlet prior with concentration parameter 0.01 on $\mathbf{\Gamma}$ (formulas for prior densities are given in Appendix D). Furthermore we expect that for some hidden states we are more likely to observe tremors than others. Therefore we specify independent Gamma distributions on state probabilities $\{p_k\}_{k=1,\dots,K}$, half of the state probabilities with mean 0.1 and variance 0.001 and the other half with mean 0.9 and variance 0.001. Also, we specify a uniform prior on hidden state means $\{\boldsymbol{\mu}^{(k)}\}_{k=1,\dots,K}$ restricted to a

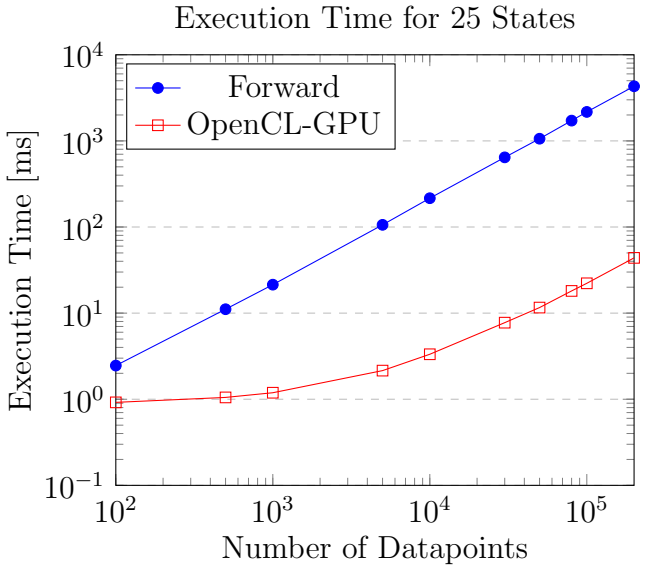


Figure 5.5: We compare computational time of OpenCL algorithm on GPU with a Forward algorithm on CPU. Computational time is indicated on the y-axis and number of datapoints are indicated by the x-axis. We see that with 10^5 datapoints, the GPU algorithm runs $\sim 10^3$ times faster.

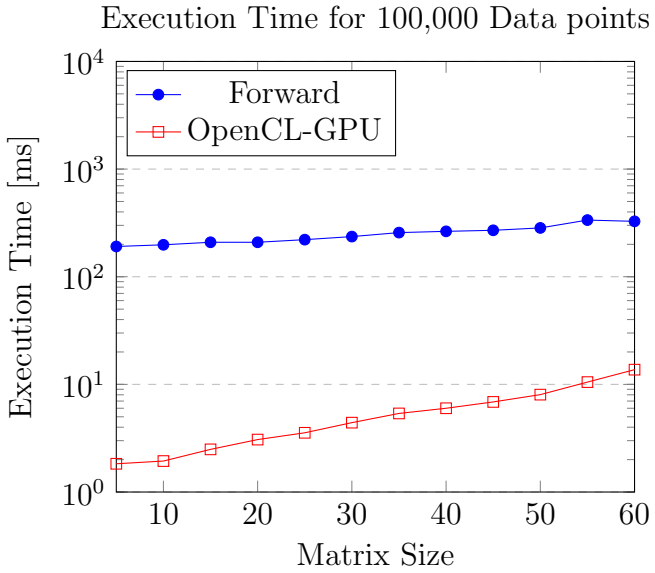


Figure 5.6: We compare computational time of OpenCL algorithm on GPU with a Forward algorithm on CPU. Computational time is indicated on the y-axis and number of HMM states are indicated by the x-axis. We see that the GPU algorithm slows down as the register capacity of compute elements is reached. However it still outperforms the Forward algorithm by orders of magnitude.

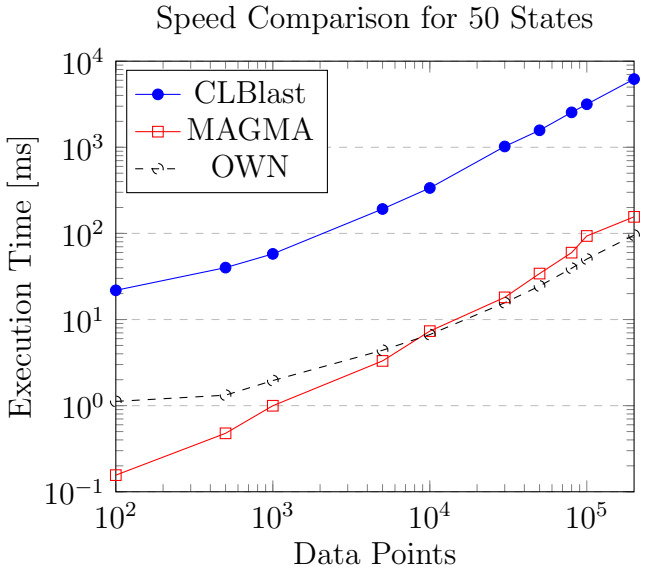


Figure 5.7: For this computational comparison (in miliseconds) with the BLAS libraries we fix the number of HMM states to $K = 50$ and increase the number of datapoints over a range of magnitude orders.

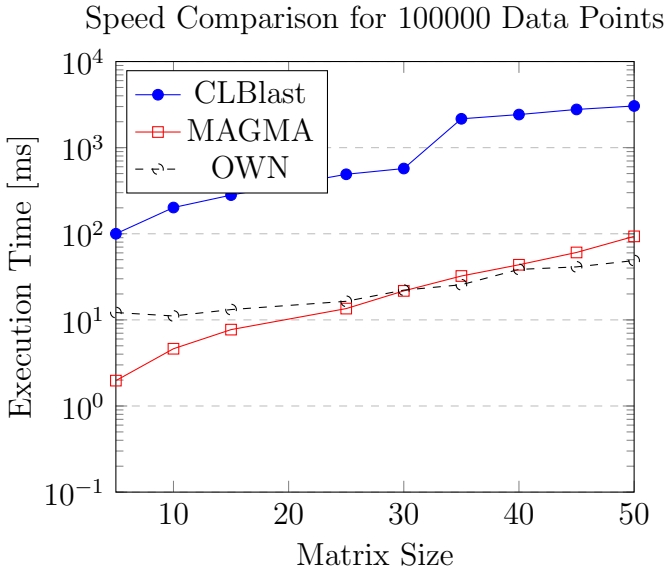


Figure 5.8: For this computational comparison (in miliseconds) with the BLAS libraries we fix the number of datapoints to $N = 100,000$ and increase the number of HMM states for $K = 5, 10, \ldots, 50$.

rectangular domain that contains all observations. We have no prior information on the shape of the hidden states therefore we specify an uninformative Inverse-Wishart prior on the covariance matrices $\{\mathbf{\Sigma}^{(k)}\}_{k=1,\ldots,K}$ with degrees of freedom equal to the number of states K and scale matrix set to a $K \times K$ identity matrix.

5.6.2 GPUeR-hmmer

In order to simulate MCMCs for the model, we incorporated the GPU likelihood algorithm along with the prior distributions into a general purpose MCMC sampler (Christen *et al.*, 2010). This Python package bundle is freely available at `https://github.com/genetica/HMMTremorRecurrencePatterns`. Note that OpenCL 1.2 and Python 3.6 (or later versions) needs to be separately installed on a system in order to support the back-end of the package. The package also contains a simple example using simulated data from the HMM described in Section 2. Additionally we provide instructions on how to modify the OpenCL code if an HMM with a different emission function is required. In order to assess convergence of the MCMC chains we used Tracer.

5.6.3 Tremor dataset of the Shikoku region

We use a large tremor dataset from the Shikoku region, Japan to demonstrate the sort of Bayesian analysis that can be done with GPUeR-hmmer. The Shikoku region is one the three major regions in Japan (the other two being the Tokai region and Kii region) in which nonvolcanic tremor occurrences have been repeatedly detected. Tremor activity spans along the strike of the Philippines Sea plate for about 600km and the depth ranges from 30 to 45 km on the plate interface. The original waveform data is supplied by the High Sensitivity Seismograph Network of the National institute for Earth Sciences and Disaster prevention in Japan. The dataset analysed by Wang *et al.* (2018) was extracted from the waveform data. It consists of $105,000$ data point measurements between 2001 and 2012. It is hourly control measurements determined using clustering and correlation methods described in Obara *et al.* (2010).

5.6.4 Model fitting

A full Bayesian analysis of the model will sample the number of hidden states along with the rest of the model parameters. However sampling from different parameter spaces is quite challenging and is an active and ongoing area of research (Lunn *et al.*,

2009). Instead we incorporate the choice of number of hidden states K into the model fitting process.

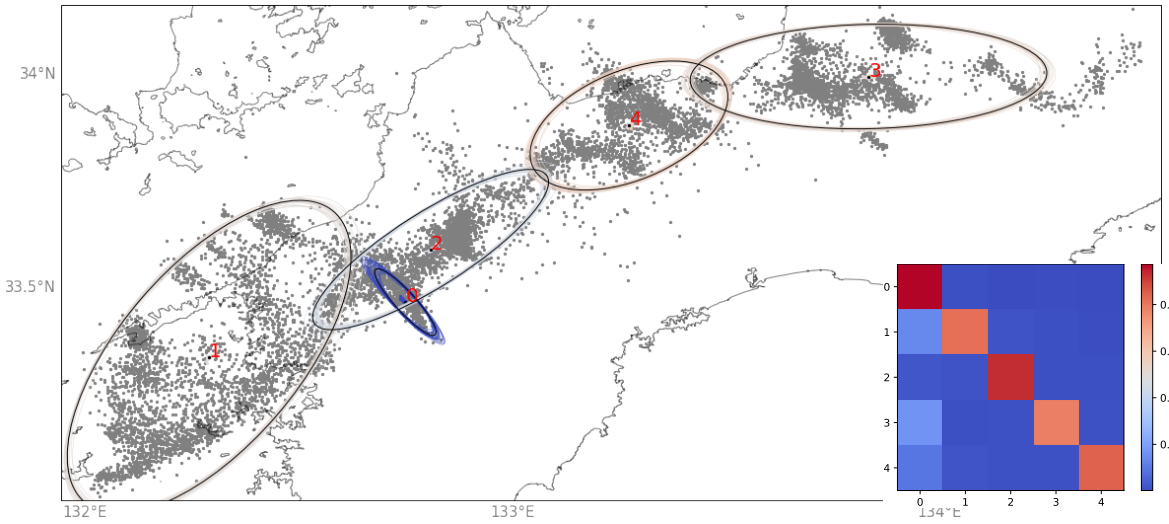
We start with a small number of hidden states and incrementally increase the number of hidden states, while doing so we assess the posterior distribution for each case. The posterior distribution of each model is estimated by running the MCMC sampler for 1,000,000 iterations. Running each chain took approximately $\sim 2-6$ hours.

In Figure 5.9 we summarize the posterior distributions for model fitted with number of hidden states $K = 5, 10, \ldots, 30$. Typically, the background states (i.e states that cover large areas) have the highest variance in posterior distribution. Whereas states covering smaller areas have considerably less variance in posterior distribution of parameters. We also see in Figure 5.9(e) that parameters used in Wang *et al.* (2018) are recovered by the posterior distribution. Typically as we increase the number of states some states are divided into two, with rare new clusters. Furthermore we see for $K = 30$ that some additional hidden states ($k = 4, 8, 26$) doesn't fit over one particular cluster of points, covers a large area, has a low probability of observing tremors and a low stationary probability (i.e time spent in state). Thereafter we also fitted models with hidden states for $K = 26, 27$ and we find that additional hidden states have the same undesirable properties and opt to use $K = 25$ as our choice for number of hidden states for the model (see MCMC summary statistics in Appendix C).

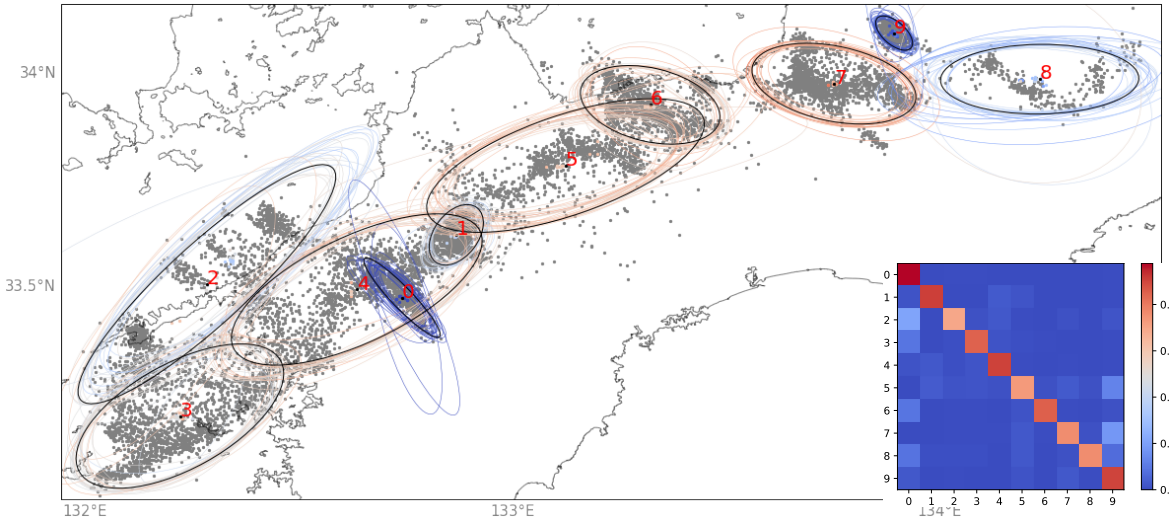
5.6.5 Forecasting

We carry out a Bayesian forecast from the model for a 5 day period (from December 11, 2012 to December 16, 2012). Note that the data for this period was excluded in the model fitting process. In order to forecast tremors we simulated 120 hourly datapoints (i.e for 5 days) from the model (with fixed number of hidden states $K = 25$) for every 1000th MCMC sample (total of 500 simulations) of the approximate posterior distribution. Note that we used the same realization of the MCMC that was generated in the model fitting process (see previous section). We used the HMM simulator in the R package *HMMextra0s* (freely available at `https://rdrr.io/cran/HMMextra0s/man/HMMextra0s-package.html`).

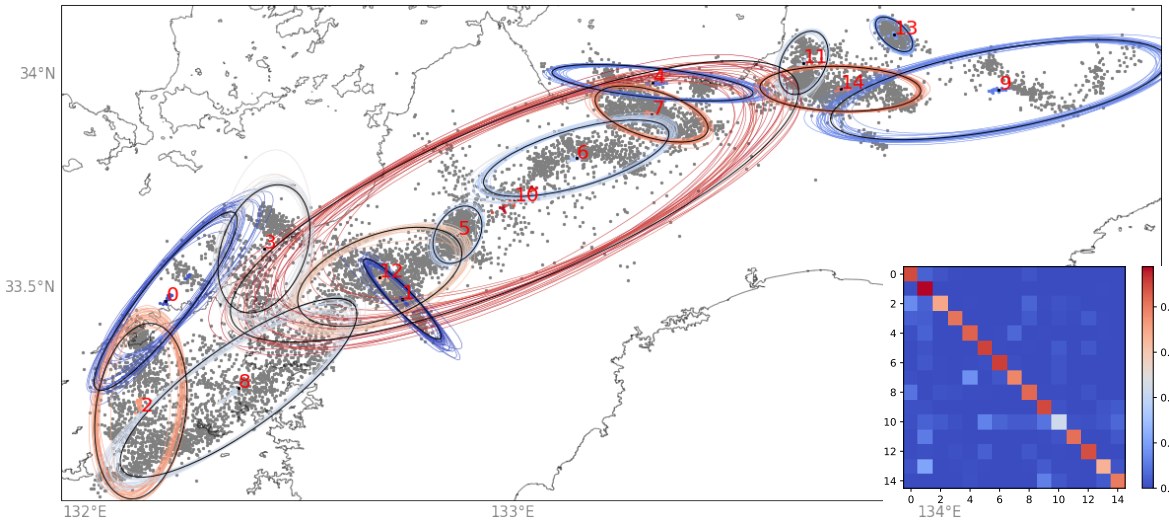
We summarize the 500 forecast simulations as a density in a longitude plot over time and a latitude plot over time (Figure 5.10). Furthermore we plot the actual data as a scatterplot using red datapoints. We also include the last day (December 10, 2012) of the data used for model fitting (as a scatterplot using black datapoints).


34°N
33.5°N
132°E
133°E
0
1
2
3
4
0
1
2
3
4
0.8
0.6
0.4
0.2

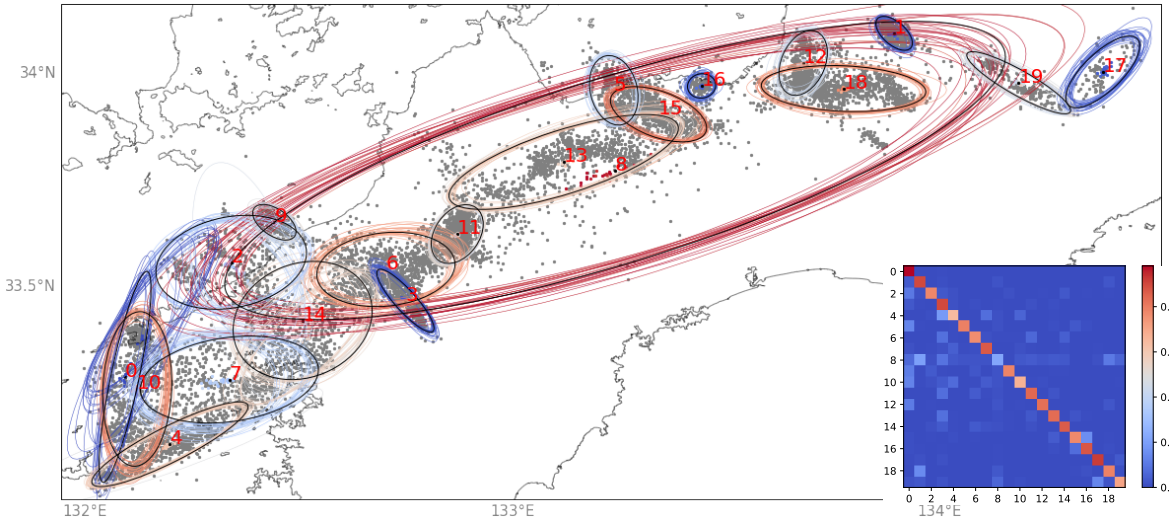
(a) $K = 5$


34°N
33.5°N
132°E
133°E
0
1
2
3
4
5
6
7
8
9
0.8
0.6
0.4
0.2
0.0

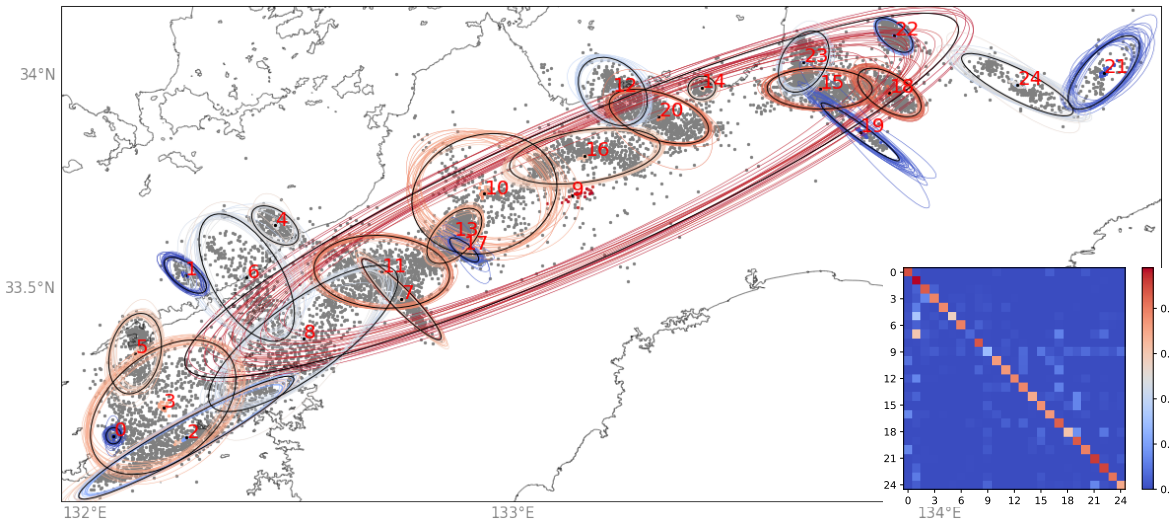
(b) $K = 10$



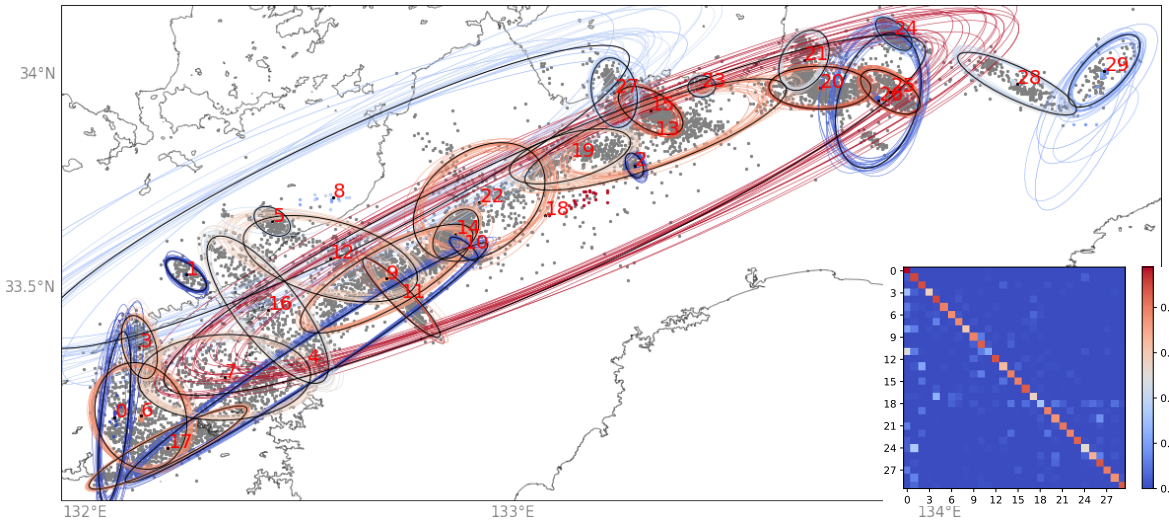
(c) $K = 15$



(d) $K = 20$

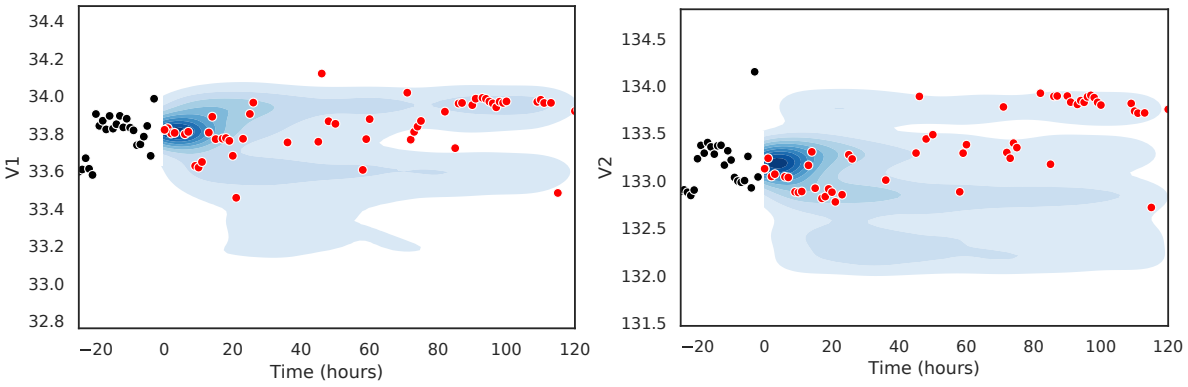


(e) $K = 25$



(f) $K = 30$

Figure 5.9: Posterior distributions of fitted models with number of hidden states $K = 5, 10, \ldots, 30$ for tremor occurrences in Shikoku region. Ellipses each map represent the 2D normal density of one hidden state for one sample from the posterior distribution. States are numbered in red. Colour of an ellipse indicate how likely a tremor will occur given the process is in the hidden state. In the bottom right corner of each map we give the mean transition matrix of the posterior distribution. Transition probabilities (array entries) and state probabilities (colour of ellipse) both use same colormap given in bottom right corner. Furthermore grey dots represent the Shikoku tremor data points. Black ellipses and -dots represent mean parameters.



(a) (b)

Figure 5.10: We summarize the forecast simulations as two (blue) density plots. (a) Latitude predictions and data plotted against time (in hours). (b) Longitude predictions and data plotted against time (in hours). The red dots in both figures are the hourly Shikoku data for the time period from December 11, 2012 to December 30, 2012 (not included in data used for model fitting). Furthermore black dots in both figures are the hourly Shikoku data for the time period December 10, 2012 (included in data used for model fitting).

We see that the model works well for the first two days. It captures nicely in which area the tremors occur. We also see that see that we get coverage from the forecast (density plot) for all the data points except for one outlier. Furthermore we see that the variance in the model predictions increases with time. This is not unexpected since the further away our forecasts are from the present the less information our data contains about the future states of the process. It would be very unlikely to make an accurate forecast of more than a week.

5.7 Discussion

In this chapter we presented an algorithm for evaluating HMM likelihoods that can run several orders of magnitude faster than the traditional Forward algorithm. The high level of parallization of the likelihood calculation translates into high data throughput.

We have implemented the algorithm for an HMM model that categorises nonvolcanic tremor data. Furthermore we have integrated the algorithm as part of an R package for Bayesian analysis using the OpenCL framework with Python under the hood. It is however expected that a CUDA implementation for NVIDIA GPUs will achieve higher data throughput but this limits the algorithm to a single vendor. OpenCL on the other hand allows execution of the algorithm on any OpenCL compliant device such as Intel CPUs, AMD CPUs and GPUs, Qualcomm processors, Xilinx FPGAs (Field-programmable gate array) and even NVIDIA GPUs.

We have reported some runtime comparisons with implementations of the Forward algorithm. The efficiency gains in computation of the likelihood allowed us to conduct a detailed Bayesian analysis for tremor data of Shikoku region of Japan.

Lastly, the OpenCL algorithm can be easily modified for other HMM models. In some cases only the evaluation function of the emission matrix needs to be updated.

5.8 Future research directions

Now that we have a better grasp of parallel algorithm design we turn back to models in phylogenetics. Specifically, we briefly answer the question: How do we implement Snapper on GPUs?

To formulate our answer we point out the main restriction to optimise data throughput for Snapper on GPUs. That is, given a species tree, to compute the likelihood at a site L_m we have that the partial likelihood at the bottom of a branch is dependent on partial likelihoods at the top of its child branches. Therefore it makes little sense to compute all partial likelihoods at a site in parallel. However we can compute partial likelihoods in parallel across many sites since we use the same species tree at each site. Using a dynamical programming approach like Snapper but with a slight twist will optimise data throughput on GPUs. To elaborate, like Snapper we start at the bottom of a species tree computing partial likelihoods as we move upwards to the root. However instead of storing partial likelihoods for just one site we store partial likelihoods for all sites as we move along the tree towards the root. This way we minimise data transfer in the memory hierarchy of the GPU. Here some work is needed to carefully layout the details of how to partition workloads on branches. Nonetheless, this simple yet powerful parallel algorithm will decrease computational time of log-likelihood evaluations by orders of magnitude and will open up avenues to even larger scale phylogenetic analysis.

References

Aerts, R. (1999). Interspecific competition in natural plant communities: mechanisms, trade-offs and plant-soil feedbacks. *Journal of experimental botany*, *50*(330), 29–37.

Bayarri, M. J. and Berger, J. O. (2004). The interplay of Bayesian and frequentist analysis. *Statistical Science*, 58–80.

Beroza, G. C. and Ide, S. (2011). Slow earthquakes and nonvolcanic tremor. *Annual review of Earth and planetary sciences*, *39*, 271–296.

Boitard, S., Rodríguez, W., Jay, F., Mona, S., and Austerlitz, F. (2016). Inferring population size history from large samples of genome-wide molecular data-an approximate Bayesian computation approach. *PLoS genetics*, *12*(3), e1005877.

Bollback, J. P., York, T. L., and Nielsen, R. (2008). Estimation of 2Nes from temporal allele frequency data. *Genetics*, *179*(1), 497–502.

Bouckaert, R., Vaughan, T. G., Barido-Sottani, J., Duchêne, S., Fourment, M., Gavryushkina, A., Heled, J., Jones, G., Kühnert, D., De Maio, N., *et al.* (2019). BEAST 2.5: An advanced software platform for Bayesian evolutionary analysis. *PLoS computational biology*, *15*(4), e1006650.

Bouckaert, R. R. (2010). DensiTree: making sense of sets of phylogenetic trees. *Bioinformatics*, *26*(10), 1372–1373.

Bryant, D., Bouckaert, R., Felsenstein, J., Rosenberg, N. A., and RoyChoudhury, A. (2012). Inferring Species Trees Directly from Biallelic Genetic Markers: Bypassing Gene Trees in a Full Coalescent Analysis. *Molecular Biology and Evolution*, *29*(8), 1917–1932.

Bryant, D. and Hahn, M. (2020). The Concatenation Question. In F. Delsuc, N. Galtier, and C. Scornavacca (Eds.), *Phylogenetics in the Genomics Era*, Chapter No.3.4, pp.3.4:1–3.4:23. A book completely handled by researchers, no publisher involved.

Chib, S. and Greenberg, E. (1995). Understanding the metropolis-hastings algorithm. *The american statistician*, *49*(4), 327–335.

Chifman, J. and Kubatko, L. (2014). Quartet inference from SNP data under the coalescent model. *Bioinformatics*, *30*(23), 3317–3324.

Chong, J., Lane, I. R., and Buthpitiya, S. W. (2014). Utilizing multiple processing units for rapid training of hidden markov models. US Patent 8,886,535.

Christen, J. A., Fox, C., *et al.* (2010). A general purpose sampling algorithm for continuous distributions (the t-walk). *Bayesian Analysis*, *5*(2), 263–281.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, *19*(90), 297–301.

Cox, J. C., Ingersoll Jr, J. E., and Ross, S. A. (2005). A theory of the term structure of interest rates. In *Theory of Valuation*, 129–164. World Scientific.

Cui, T., Fox, C., and O'sullivan, M. (2011). Bayesian calibration of a large-scale geothermal reservoir model by a new adaptive delayed acceptance Metropolis Hastings algorithm. *Water Resources Research*, *47*(10).

Cui, T., Fox, C., and O'Sullivan, M. J. (2019). A posteriori stochastic correction of reduced models in delayed-acceptance MCMC, with application to multiphase subsurface inverse problems. *International Journal for Numerical Methods in Engineering*, *118*(10), 578–605.

Cuzzocrea, A., Mumolo, E., Timeus, N., and Vercelli, G. (2016). GPU-Aware Genetic Estimation of Hidden Markov Models for Workload Classification Problems. *Proceedings - International Computer Software and Applications Conference*, *1*, 674–682.

Degnan, J. H. and Rosenberg, N. A. (2009). Gene tree discordance, phylogenetic inference and the multispecies coalescent. *Trends in ecology & evolution*, *24*(6), 332–340.

Dormand, J. R. and Prince, P. J. (1980). A family of embedded Runge-Kutta formulae. *Journal of computational and applied mathematics*, *6*(1), 19–26.

Dormann, C. F., Schymanski, S. J., Cabral, J., Chuine, I., Graham, C., Hartig, F., Kearney, M., Morin, X., Römermann, C., Schröder, B., *et al.* (2012). Correlation

and process in species distribution models: bridging a dichotomy. *Journal of Biogeography*, *39*(12), 2119–2131.

Douglas, Jr, J. and Dupont, T. (1970). Galerkin methods for parabolic equations. *SIAM Journal on Numerical Analysis*, *7*(4), 575–626.

Eaton, D. A. and Overcast, I. (2016). ipyrad: Interactive assembly and analysis of RADseq datasets. *Bioinformatics*.

Edwards, S. V. (2009). Is a new and general theory of molecular systematics emerging? *Evolution: International Journal of Organic Evolution*, *63*(1), 1–19.

Epstein, C. L. and Mazzeo, R. (2010). Wright–Fisher diffusion in one dimension. *SIAM Journal on Mathematical Analysis*, *42*(2), 568–608.

Etheridge, A. (2011). *Some mathematical models from population genetics : École d'éte de probabilités De Saint-flour XXXIX-2009*. Springer.

Ethier, S. N. and Kurtz, T. G. (1986). *Markov processes : characterization and convergence*. Wiley.

Ethier, S. N. and Norman, M. F. (1977). Error estimate for the diffusion approximation of the Wright–Fisher model. *Proceedings of the National Academy of Sciences*, *74*(11), 5096–5098.

Evans, L. C. (2010). *Partial differential equations*, Volume 19. American Mathematical Soc.

Ewens, W. (2004). *Mathematial population genetics. I. Theoretical introduction* (2 ed.). Interdisciplinary Applied Mathematics. New York: Springer.

Felsenstein, J. (1981). Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, *17*(6), 368–376.

Felsenstein, J. (1988). Phylogenies and quantitative characters. *Annual Review of Ecology and Systematics*, *19*(1), 445–471.

Felsenstein, J. (1992). Phylogenies from restriction sites: a maximum-likelihood approach. *Evolution*, *46*(1), 159–173.

Felsenstein, J. (2004). *Inferring phylogenies*, Volume 2. Sinauer associates Sunderland, MA.

Fisher, R. A. (1930). *The genetical theory of natural selection*. Clarendon Press.

Forbes, C., Evans, M., Hastings, N., and Peacock, B. (2011). *Statistical distributions*. John Wiley & Sons.

Fox, L. and Parker, I. (1968). *Chebyshev Polynomials in Numerical Analysis*. Oxford Mathematical Handbooks. London: Oxford University Press.

Gelfand, A. E. and Smith, A. F. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, *85*(410), 398–409.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall/CRC.

Georges, A., Gruber, B., Pauly, G. B., White, D., Adams, M., Young, M. J., Kilian, A., Zhang, X., Shaffer, H. B., and Unmack, P. J. (2018). Genomewide SNP markers breathe new life into phylogeography and species delimitation for the problematic short-necked turtles (Chelidae: Emydura) of eastern Australia. *Molecular Ecology*, *27*(24), 5195–5213.

Golub, G. H. and Van Loan, C. (2013). Matrix computations 4th Edition.

Griffiths, R. C. and Spano‘, D. (2010). Diffusion processes and the coalescent. In N. H. Bingham and C. M. Goldie (Eds.), *Probability and Mathematical Genetics: Papers in Honour of Sir John Kingman*, London Mathematical Society Lecture Note Series, 358–379. Cambridge University Press.

Gutenkunst, R., Hernandez, R., Williamson, S., and Bustamante, C. (2010). Diffusion approximations for demographic inference: DaDi. *Nature Precedings*, *5*, 1–1.

Gutenkunst, R. N., Hernandez, R. D., Williamson, S. H., and Bustamante, C. D. (2009). Inferring the joint demographic history of multiple populations from multidimensional SNP frequency data. *PLoS Genetics*, *5*(10).

Gutteridge, P., Voice, C., Stones, D., and Haig, A. (1988). Buddy. Recorded: January-June 1988 at Mascot Studios, auckland, NZ. Engineered by Victor Grbic and Terry Moore. Mixed by Victor Gbric and Snapper.

Hein, J., Schierup, M., and Wiuf, C. (2004). *Gene genealogies, variation and evolution: a primer in coalescent theory*. Oxford University Press, USA.

Higgins, S. I., O'Hara, R. B., Bykova, O., Cramer, M. D., Chuine, I., Gerstner, E.-M., Hickler, T., Morin, X., Kearney, M. R., Midgley, G. F., *et al.* (2012). A physiological analogy of the niche for projecting the potential distribution of plants. *Journal of Biogeography*, *39*(12), 2132–2145.

Hiscott, G., Fox, C., Parry, M., and Bryant, D. (2016). Efficient Recycled Algorithms for Quantitative Trait Models on Phylogenies. *Genome biology and evolution*, *8*(5), 1338–1350.

Horn, D. R., Houston, M., and Hanrahan, P. (2005). ClawHMMER: A streaming HMMer-search implementation. *Proceedings of the ACM/IEEE 2005 Supercomputing Conference, SC'05*, *2005*.

Huelsenbeck, J. P. and Rannala, B. (2004). Frequentist properties of Bayesian posterior probabilities of phylogenetic trees under simple and complex substitution models. *Systematic biology*, *53*(6), 904–913.

Iserles, A. (2009). *A first course in the numerical analysis of differential equations*. Number 44. Cambridge university press.

Jenkins, P. A., Spano, D., *et al.* (2017). Exact simulation of the Wright–Fisher diffusion. *The Annals of Applied Probability*, *27*(3), 1478–1509.

Kearney, M. and Porter, W. (2009). Mechanistic niche modelling: combining physiological and spatial data to predict species' ranges. *Ecology letters*, *12*(4), 334–350.

Kindratenko, V. (2014). *Numerical computations with GPUs*. Springer.

Kingman, J. (1982a). The coalescent. *Stochastic Processes and their Applications*, *13*(3), 235–248.

Kingman, J. F. (2000). Origins of the coalescent: 1974-1982. *Genetics*, *156*(4), 1461–1463.

Kingman, J. F. C. (1982b). The coalescent. *Stochastic processes and their applications*, *13*(3), 235–248.

Lapierre, M., Lambert, A., and Achaz, G. (2017). Accuracy of demographic inferences from the site frequency spectrum: the case of the Yoruba population. *Genetics*, *206*(1), 439–449.

Lepage, T., Lawi, S., Tupper, P., and Bryant, D. (2006). Continuous and tractable models for the variation of evolutionary rates. *Mathematical biosciences*, *199*(2), 216–233.

Leslie, A. B., Beaulieu, J. M., Rai, H. S., Crane, P. R., Donoghue, M. J., and Mathews, S. (2012). Hemisphere-scale differences in conifer evolutionary dynamics. *Proceedings of the National Academy of Sciences*, *109*(40), 16217–16221.

Li, J., Chen, S., and Li, Y. (2009). The fast evaluation of hidden Markov models on GPU. *Proceedings - 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS 2009*, *4*(60802026), 426–430.

Liu, C. (2009). cuHMM: a CUDA implementation of hidden Markov model training and classification. *The Chronicle of Higher Education*, 1–13.

Liu, L. (2008). BEST: Bayesian estimation of species trees under the coalescent model. *Bioinformatics*, *24*(21), 2542–2543.

Liu, L., Pearl, D. K., Brumfield, R. T., and Edwards, S. V. (2008). Estimating species trees using multiple-allele DNA sequence data. *Evolution: International Journal of Organic Evolution*, *62*(8), 2080–2091.

Liu, L., Yu, L., and Edwards, S. V. (2010). A maximum pseudo-likelihood approach for estimating species trees under the coalescent model. *BMC Evolutionary Biology*, *10*(1), 302.

Lukic, S. and Hey, J. (2012). Demographic Inference Using Spectral Methods on SNP Data, with an Analysis of the Human Out-of-Africa Expansion. *Genetics*, *192*(2), 619–639.

Lunn, D. J., Best, N., and Whittaker, J. C. (2009). Generic reversible jump MCMC using graphical models. *Statistics and Computing*, *19*(4), 395.

MacArthur, R. H. (1984). *Geographical ecology: patterns in the distribution of species*. Princeton University Press.

Mason, J. C. and Handscomb, D. C. (2002). *Chebyshev polynomials*. Chapman and Hall/CRC.

McKane, A. J. and Waxman, D. (2007). Singular solutions of the diffusion equation of population genetics. *Journal of Theoretical Biology*, *247*(4), 849–858.

Meier, E. S., Kienast, F., Pearman, P. B., Svenning, J.-C., Thuiller, W., Araújo, M. B., Guisan, A., and Zimmermann, N. E. (2010). Biotic and abiotic variables show little redundancy in explaining tree species distributions. *Ecography*, *33*(6), 1038–1048.

Mod, H. K., Scherrer, D., Luoto, M., and Guisan, A. (2016). What we use is not what we know: environmental predictors in plant distribution models. *Journal of Vegetation Science*, *27*(6), 1308–1322.

Moran, P. A. P. (1958). Random processes in genetics. In *Mathematical proceedings of the cambridge philosophical society*, Volume 54, 60–71. Cambridge University Press.

Nagylaki, T. (1989). The maintenance of genetic variability in two-locus models of stabilizing selection. *Genetics*, *122*(1), 235–248.

Nee, S. (2001). Inferring speciation rates from phylogenies. *Evolution*, *55*(4), 661–668.

Nielsen, R. (2000). Estimation of population parameters and recombination rates from single nucleotide polymorphisms. *Genetics*, *154*(2), 931–942.

Obara, K., Tanaka, S., Maeda, T., and Matsuzawa, T. (2010). Depth-dependent activity of non-volcanic tremor in southwest Japan. *Geophysical Research Letters*, *37*(13).

Øksendal, B. (2003). *Stochastic differential equations, an Introduction with Applications*. Universitext. Springer.

Pazy, A. (2012). *Semigroups of linear operators and applications to partial differential equations*, Volume 44. Springer Science & Business Media.

Piessens, R., de Doncker-Kapenga, E., Überhuber, C. W., and Kahaner, D. K. (2012). *Quadpack: a subroutine package for automatic integration*, Volume 1. Springer Science & Business Media.

Press, S. J. (2005). *Applied multivariate analysis: using Bayesian and frequentist methods of inference*. Courier Corporation.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Racimo, F., Renaud, G., and Slatkin, M. (2016). Joint estimation of contamination, error and demography for nuclear DNA from ancient humans. *PLoS genetics*, *12*(4), e1005972.

Rambaut, A., Drummond, A. J., Xie, D., Baele, G., and Suchard, M. A. (2018). Posterior summarization in Bayesian phylogenetics using Tracer 1.7. *Systematic Biology*, *67*(5), 901–904.

Rannala, B. and Yang, Z. (2017). Efficient Bayesian species tree inference under the multispecies coalescent. *Systematic biology*, *66*(5), 823–842.

Raue, A., Kreutz, C., Theis, F. J., and Timmer, J. (2013). Joining forces of Bayesian and frequentist methodology: a study for inference in the presence of non-identifiability. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *371*(1984), 20110544.

Schimper, A. (1903). Plant geography upon a physiological basis. Clarendon.

Schmelzer, T. and Trefethen, L. N. (2007). Evaluating matrix functions for exponential integrators via Carathéodory-Fejér approximation and contour integrals. *Electronic Transactions on Numerical Analysis*, *29*, 1–18.

Shampine, L. F. (1986). Some practical runge-kutta formulas. *Mathematics of Computation*, *46*(173), 135–150.

Sirén, J., Marttinen, P., and Corander, J. (2010). Reconstructing population histories from single nucleotide polymorphism data. *Molecular Biology and Evolution*, *28*(1), 673–683.

Song, S., Liu, L., Edwards, S. V., and Wu, S. (2012). Resolving conflict in eutherian mammal phylogeny using phylogenomics and the multispecies coalescent model. *Proceedings of the National Academy of Sciences*, *109*(37), 14942–14947.

Song, Y. S. and Steinrücken, M. (2012). A simple method for finding explicit analytic transition densities of diffusion processes with general diploid selection. *Genetics*, *190*(3), 1117–29.

Stakgold, I. and Holst, M. J. (2011). *Green's functions and boundary value problems*, Volume 99. John Wiley & Sons.

Steinrücken, M., Bhaskar, A., and Song, Y. S. (2014). A novel spectral method for inferring general diploid selection from time series genetic data. *The Annals of Applied Statistics*, *8*(4), 2203–2222.

Tataru, P., Simonsen, M., Bataillon, T., and Hobolth, A. (2017). Statistical inference in the Wright–Fisher model using allele frequency data. *Systematic biology*, *66*(1), e30–e46.

Thornley, J. H. (1995). Shoot: root allocation with respect to C, N and P: an investigation and comparison of resistance and teleonomic models. *Annals of Botany*, *75*(4), 391–405.

Tilman, D. (1988). *Plant strategies and the dynamics and structure of plant communities*. Princeton University Press.

Trefethen, L. N. (2000). *Spectral methods in MATLAB*, Volume 10. Siam.

Trefethen, L. N. (2013). *Approximation theory and approximation practice*. Philadelphia: SIAM.

Vachaspati, P. and Warnow, T. (2015). ASTRID: accurate species trees from internode distances. *BMC Genomics*, *16*(10), S3.

Waldvogel, J. (2006). Fast construction of the Fejér and Clenshaw–Curtis quadrature rules. *BIT Numerical Mathematics*, *46*(1), 195–202.

Wang, T., Zhuang, J., Buckby, J., Obara, K., and Tsuruoka, H. (2018). Identifying the Recurrence Patterns of Nonvolcanic Tremors Using a 2-D Hidden Markov Model With Extra Zeros. *Journal of Geophysical Research: Solid Earth*, *123*(8), 6802–6825.

Williamson, S. H., Hernandez, R., Fledel-Alon, A., Zhu, L., Nielsen, R., and Bustamante, C. D. (2005). Simultaneous inference of selection and population growth from patterns of variation in the human genome. *Proceedings of the National Academy of Sciences*, *102*(22), 7882–7887.

Wright, S. (1931). Evolution in Mendelian populations. *Genetics*, *16*(2), 97.

Yang, Z. (2015). The BPP program for species tree estimation and species delimitation. *Current Zoology*, *61*(5), 854–865.

Yao, P., An, H., Xu, M., Liu, G., Li, X., Wang, Y., and Han, W. (2010). CuHMMer: A load-balanced CPU-GPU cooperative bioinformatics application. *Proceedings of the 2010 International Conference on High Performance Computing and Simulation, HPCS 2010*, 24–30.

Yu, L., Ukidave, Y., and Kaeli, D. (2015). GPU-Accelerated HMM for Speech Recognition. *Proceedings of the International Conference on Parallel Processing Workshops*, *2015-May*, 395–402.

Yule, G. U. (1925). Ii.—a mathematical theory of evolution, based on the conclusions of dr. jc willis, fr s. *Philosophical transactions of the Royal Society of London. Series B, containing papers of a biological character*, *213*(402-410), 21–87.

Zhang, D., Zhao, R., Han, L., Wang, T., and Qu, J. (2009). An implementation of viterbi algorithm on GPU. *2009 1st International Conference on Information Science and Engineering, ICISE 2009*, 121–124.

Zucchini, W., MacDonald, I. L., and Langrock, R. (2017). *Hidden Markov models for time series: an introduction using R*. Chapman and Hall/CRC.

Appendix A

Genetic glossary

Allele A variation of a gene in the genome sequence of a species.

Chromosome Genetic material that contain a partial genome sequence.

Diploid An organism with two sets of chromosomes, one set of chromosomes is inherited paternally, the other set is inherited maternally.

Gene A collection of nucleotides that encode a set of instructions that form a basic functional unit of heredity.

Genealogy The whole family tree structure.

Genetic drift Fluctuation in allele frequency in a population due to reproduction.

Gene trees Evolutionary history of a gene.

Genome sequence An ordered list of nucleotides that encodes the complete set of inheritable instructions.

Genome Genetic material that contains the genome sequence.

Haploid An organism with one set of chromosomes.

Locus Position of a gene in a genome sequence.

Nucleotide An organic molecule that functions as the basic unit for encoding inheritable instructions of an organism.

Recombination An exchange of genetic material between reproducing individuals which leads to offspring with combination of genes that differ from those found in either parent.

Selection Change in frequency of an allele due to influence on it own chance of being passed on to the next generation.

Single nucleotide polymorphism A substitution of a nucleotide at a specific site in the genome sequence.

Site The position of a nucleotide in the genome sequence.

Species tree Evolutionary history of a species. Here we assume a species tree is a rooted bifurcating tree.

Appendix B

Dealing with boundary conditions

B.1 Specifying boundaries

In the sections that follow we answer the following question: Given that we know the form, regularity and boundary conditions of the forward diffusion, what should the boundary conditions be on the backward diffusion such that there exist a unique operator that is adjoint to the forward diffusion. To answer this properly we consider two cases, the first case considers only smooth functions and then we slightly generalize to the case when we have smooth solutions and point masses (non-smooth solutions) where point masses are restricted to the boundary.

B.2 Adjoints and boundary conditions: density case

In Chapter 2 we have a time-homogeneous, continuous Markov process X_t with state space $x \in [0, 1]$ and $t \in [0, T]$. Suppose (to begin with) that the process is well-behaved enough that there is a density $f(x, t)$ for each t. That is, for all t,

$$Pr[a < X_t < b] = \int_a^b f(x, t)dx.$$

Let μ be the density for X_0. We suppose that f satisfies

$$\frac{d}{dt}f(x, t) = \mathcal{L}^* f(x, t)$$

where

$$\mathcal{L}^* = -\frac{\partial}{\partial x}\left(\beta_2(1-x) - \beta_1 x\right) + \frac{1}{2}\frac{\partial^2}{\partial x^2}x(1-x)$$

together with the constraints that $f(0,x) = \mu(x)$ for all x, that $f(x,t)$ is a density for all t, and that there is no mechanisms under which X_t might jump instantaneously from 0 to 1 or back.

The constraint that mass is preserved gives

$$\frac{d}{dt}\int_0^1 f(x,t)dx = 0$$

which, on substituting the backward diffusion, gives

$$\frac{1}{2}\frac{\partial}{\partial x}(b(x)p(x,t)) - a(x)f(x,t)\Big|_{x=1} - \frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\Big|_{x=0} = 0. \quad \text{(B.1)}$$

The assumption that the process can't jump from 0 to 1 then gives

$$\frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\Big|_{x=1} = \frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\Big|_{x=0} = 0.$$

In order to determine the adjoint $\mathcal{L}$, see Stakgold and Holst (2011, pg. 317) for more on the adjoint of an unbounded operator, and to find out which boundary conditions g has to satisfy, we start with the equation

$$\langle -\mathcal{L}^* f(x,t), g(x,t)\rangle = \langle f(x,t), \mathcal{L}g(x,t)\rangle$$

using integration by parts, and substituting in (weaker) boundary condition (B.1):

$$\begin{aligned}
\langle \mathcal{L}^* f(x,t), g(x,t)\rangle &= -\int_0^1 \frac{\partial}{\partial x}\left(\frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\right) g(x,t)dx \\
&= \int_0^1 \left(\frac{1}{2}b(x)\frac{\partial}{\partial x}f(x,t) - a(x)f(x,t)\right)\frac{\partial}{\partial x}g(x,t)dx \\
&+ \left(\frac{1}{2}b(x)\frac{\partial}{\partial x}f(x,t) - a(x)f(x,t)\right) g(x,t)\Big|_{x=0}^1 \\
&= \int_0^1 \frac{1}{2}b(x)\frac{\partial}{\partial x}f(x,t)\frac{\partial}{\partial x}g(x,t)dx - \int_0^1 a(x)f(x,t)\frac{\partial}{\partial x}g(x,t)dx \\
&= \int_0^1 f(x,t)\left(\frac{1}{2}b(x)\frac{\partial^2}{\partial x^2}g(x,t)dx + a(x)\frac{\partial}{\partial x}g(x,t)\right) \\
&- f(x,t)\frac{1}{2}b(x)\frac{\partial}{\partial x}g(x,t)\Big|_{x=0}^1 \\
&= \langle f(x,t), \mathcal{L}g(x,t)\rangle - \frac{1}{2}b(x)f(x,t)\frac{\partial}{\partial x}g(x,t)\Big|_{x=0}^1 .
\end{aligned}$$

We find that

$$\mathcal{L}g(x,t) = b(x)\frac{\partial^2}{\partial x^2}g(x,t) - a(x)\frac{\partial}{\partial x}g(x,t)$$

with boundary condition

$$\left[b(x)f(x,t)\frac{\partial}{\partial x}g(x,t)\right]_0^1 = 0$$

implying (with the assumption of independence at $0, 1$) the boundary condition

$$\left[b(x)\frac{\partial}{\partial x}g(x,t)\right]_{x=0,1} = 0.$$

Given an integrable function h we define

$$g(x,t) = E(h(X_T)|X_{T-t} = x) = \int_0^1 h(y)f(y,t)dy.$$

It follows from the definition of unconditional expectation that

$$\int_0^1 g(x,t)f(x,t)dx = E(h(X_T)).$$

Now taking the derivative in terms of t on both sides. Note that $E(h(X_T))$ is independent of t therefore the LHS gives us

$$\frac{\partial}{\partial t}E(h(X_T)) = 0.$$

Taking the derivative of t on the RHS we have

$$\begin{aligned}
\frac{\partial}{\partial t}\langle g(x,t), f(x,t)\rangle &= \left\langle \frac{\partial}{\partial t}g(x,t), f(x,t) \right\rangle + \left\langle g(x,t), \frac{\partial}{\partial t}f(x,t) \right\rangle \\
&= \left\langle \frac{\partial}{\partial t}g(x,t), f(x,t) \right\rangle + \langle g(x,t), \mathcal{L}^* f(x,t)\rangle \\
&= \left\langle \frac{\partial}{\partial t}g(x,t), f(x,t) \right\rangle - \langle \mathcal{L}g(x,t), f(x,t)\rangle \\
&= \left\langle \frac{\partial}{\partial t}g(x,t) - \mathcal{L}g(x,t), f(x,t) \right\rangle.
\end{aligned}$$

Since

$$0 = \left\langle \frac{\partial}{\partial t}g(x,t) - \mathcal{L}g(x,t), f(x,t) \right\rangle$$

holds for any f we conclude that

$$\frac{\partial}{\partial t}g(x,t) = \mathcal{L}g(x,t).$$

The adjoint boundary conditions appears to be satisfied when $b(x) = x(1-x)$ and assuming $g(x,t)$ is smooth. However we would like to establish the smoothness of $g(x,t)$ rather than just assume it. The regularity (smoothness) of $g(x,t)$ is not only important in theory. It is also important in practise. We will use the forward diffusion for computing likelihoods. The smoothness of $g(x,t)$ is important for the choice of numerical approximation method in Chapter 3 since it is much easier to approximate smooth solutions than it is non-smooth solutions.

Therefore we have to establish uniqueness and regularity for solutions $g(x,t)$. We first prove uniqueness by again assuming regularity on $g(x,t)$ and applying a Maximum Principle argument (Evans, 2010). This also gives us an additional boundary condition on $g(x,t)$ that needs to be satisfied. Thereafter we state a result that ensures that our

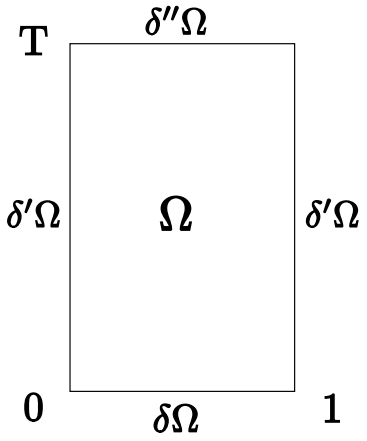


Figure B.1: Boundary labels for Ω in Proposition B.2.1

solutions $g(x,t)$ to the forward diffusion is smooth if the initial condition is smooth.

Proposition B.2.1. *Let $a(0), -a(1) > 0$ and suppose $g(x,t)$ satisfies*

$$\frac{d}{dt}g(x,t) - \mathcal{L}g(x,t) \leq 0 \tag{B.2}$$

then $g(x,t)$ is a unique solution if:

1. *We have $g(x,t) \in \mathcal{C}^2_{[0,1]}$ with respect to x and $g(x,t) \in \mathcal{C}^1_{(0,\infty)}$ with respect to t.*
2. *The following boundary condition hold*

$$\left[b(x)\frac{\partial^2}{\partial x^2}g(x,t)\right]_{x=0,1} = 0.$$

Proof (Epstein and Mazzeo, 2010). Suppose we have $g(x,t) \in \mathcal{C}^2_{[0,1]}$ with respect to x and $g(x,t) \in \mathcal{C}^1_{(0,\infty)}$ with respect to t. For some $\epsilon > 0$ define a sup solution

$$g_\epsilon(x,t) = g(x,t) + \frac{\epsilon}{1+t}$$

such that

$$\frac{\partial}{\partial t}g_\epsilon - \mathcal{L}g_\epsilon(x,t) < 0. \tag{B.3}$$

Suppose that $g(x^*,t^*) = \max(g(x,t))$ occurs somewhere on the interior of the domain $\bar{\Omega} = [0,1]\times[0,T]$. This leads to a contradiction since $\frac{\partial}{\partial t}g(t^*,x^*) = 0$, $\frac{\partial}{\partial x}g(t^*,x^*) = 0$ and $\frac{\partial^2}{\partial x^2}g(t^*,x^*) \leq 0$ applied to the inequality (B.3) implies that g_ϵ is not a sup solution.

The same argument holds if $g(x^*, t^*)$ is on $\delta''\Omega$ (See Figure B.1) since $\frac{\partial}{\partial t}g(T, x^*) \geq 0, \frac{\partial}{\partial x}g(T, x^*) = 0$ and $\frac{\partial^2}{\partial x^2}g(T, x^*) \leq 0$. Now suppose $g(x^*, t^*)$ is on $\delta'\Omega$ (See Figure B.1). This will also violate the assumption that g_ϵ is a supsolution. At $x = 0$ we have

$$\frac{d}{dt}g_\epsilon(0, t^*) - \mathcal{L}g_\epsilon(0, t^*) \leq 0$$

this leads to

$$-\epsilon/(1 + t^*)^2 + K \leq 0$$

for $K \geq 0$ since we have $a(0) > 0$ and $\frac{\partial}{\partial t}g(0, t^*) = 0$, $\frac{\partial}{\partial x}g(0, t^*) \leq 0$ and $\frac{\partial^2}{\partial x^2}g(0, t^*) = 0$. The contradiction follows if we choose $\epsilon > K(1 + t^*)^2$. Same argument holds for the case $x = 1$. We therefore have that

$$\max(g(x, t)) < \max(g(x, 0)) + \epsilon$$

since this holds for any ϵ, we conclude that the maximum can only occur at the boundary with respect to t, that is

$$\max(g(x, t)) = \max(g(x, 0)).$$

Now uniqueness follows from the following argument. Suppose that g, u are both solutions that satisfy the backward diffusion and the boundary conditions. Then $w = g - u$ is also a solution due to the linearity. We have shown that the maximum can only occur on $\delta\Omega$. Since the initial condition for w is $w(x, 0) = 0$ we have

$$\max(w) = \max(-w) = 0$$

on $\bar{\Omega}$. Therefore $g = u$ on $\bar{\Omega}$. □

This boundary conditions also appears to be satisfied automatically when $b(x)$ is of the form $x(1 - x)$ and g has bounded derivatives.

If the initial condition is smooth then the below result in Epstein and Mazzeo (2010) gives us the sought after uniqueness and regularity for the general case of the backwards diffusion albeit after 40 pages of convincing. Epstein and Mazzeo (2010) construct the RHS of the backward diffusion in terms of Green's functions, also called heat kernels, in doing so they get an analytical handle on the behaviour of the solutions. It then

follows from the properties of the kernels that there exist a semigroup that describes the dynamics of solutions given some initial condition. Semigroups are algebraic structures consisting of a set together with an associative binary operation. More specifically, a linear operator T on a Banach space (complete normed function space) is a *semigroup* if (i) $T(s+t) = T(s)T(t)$ (ii) $T(0) = I$. Semigroups are mathematically convenient to work with and have some established regularity, uniqueness and existence results, see Pazy (2012) for a detailed treatment. In the Theorem that follows semigroups show up on the peripheral but we will not use semigroups in any rigorous way.

Theorem B.2.2 ((Epstein and Mazzeo, 2010) pg. 595)**.** *For each $m \in \mathbb{N} \cup \{0\}$ the operator $T(t)$ define a semigroup on $\mathcal{C}^m_{[0,1]}$. The function $g(x,t) = T(t)g(x,0)$ satisfies,*

$$\frac{\partial}{\partial t}g(x,t) = b(x)\frac{\partial^2}{\partial x^2}g(x,t) + a(x)\frac{\partial}{\partial x}g(x,t)$$

where $a(0), a(-1) > 0$. Moreover, for $g(x,0) \in \mathcal{C}^m_{[0,1]}$

$$\lim_{t \to +} \|T(t)g(x,0) - g(x,0)\| = 0$$

with respect to a norm $\|.\|$ defined on $C^m_{[0,1]}$.

Choosing our initial condition such that $g(x,0) \in C^2_{[0,1]}$ will ensure that $g(x,t)$ is smooth enough to satisfy the adjoint boundary condition as well as the uniqueness boundary condition (Theorem B.2.2). We note that initial conditions we work with in Chapter 2 are polynomials of the form

$$g(x,0) = \binom{n}{r} x^r (1-x)^{n-r}.$$

B.3 Adjoint and boundary conditions: general case

We now suppose that

$$Pr[a < X_t < b] = \int_a^b f(x,t)dx$$

but that there are also point masses at 0 and 1:

$$Pr[X_t = 0] = f(0,t), \qquad Pr[X_t = 1] = f(1,t).$$

Point masses on the boundary are important to consider when either the forward mutation rate (β_1) or backward mutation rate (β_2) are zero. If there is no mutation then allele frequencies can fixate leading to mass accumulating on the boundary (that is point mass).

We suppose that f satisfies

$$\frac{\partial}{\partial t} f(x,t) = \mathcal{L}^* f(x,t)$$

together with the constraints that $f(0,x) = \mu(x)$ for all x.

We have

$$\frac{\partial}{\partial t}\left[\int_0^1 f(x,t)dx + f(0,t) + f(1,t)\right] = 0$$

which, as above and with the independence conditions, gives

$$\begin{aligned}
\frac{\partial}{\partial t} f(0,t) &= \frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\Big|_{x=0}, \\
\frac{\partial}{\partial t} f(1,t) &= -\frac{1}{2}\frac{\partial}{\partial x}(b(x)f(x,t)) - a(x)f(x,t)\Big|_{x=1}.
\end{aligned}$$

Given a function h we define

$$g(x,t) = E[h(X_T)|X_{T-t} = x] = \int_0^1 f(y,t)h(y)dy + f(0,t)h(0) + f(1,t)h(1).$$

From the definition of unconditional expectation we have

$$\int_0^1 f(x,t)g(x,t)dx + f(0,t)g(0,t) + f(1,t)g(1,t) = E[h(X_T)]$$

is constant as a function of t, so that

$$\frac{\partial}{\partial t}\left[\int_0^1 f(x,t)g(x,t)dy + f(0,t)g(0,t) + f(1,t)g(1,t)\right] = 0$$

which implies for all t we have

$$\int_0^1 \left(\frac{\partial}{\partial t} f(x,t)\right) g(x,t)dx + \frac{\partial}{\partial t} f(0,t)g(0,t) + \frac{\partial}{\partial t} f(1,t)g(1,t)$$

$$= -\int_0^1 f(x,t)\left(\frac{\partial}{\partial t}g(x,t)\right)dx - f(0,t)\frac{\partial}{\partial t}g(0,t) - f(1,t)\frac{\partial}{\partial t}g(1,t).$$

Let $\langle , \rangle$ denote the real inner product

$$\langle f, g\rangle = \int_0^1 f(x)g(x)dx + f(0)g(0) + f(1)g(1).$$

We have that for all g satisfying the PDE (1.6) and boundary conditions, and all f,

$$\begin{aligned}\left\langle f(x,t), \left(\frac{\partial}{\partial t}g(x,t)\right)\right\rangle &= \left\langle \left(\frac{\partial}{\partial t}f(x,t)\right), g(x,t)\right\rangle \\ &= \langle \mathcal{L}^* f(x,t), g(x,t)\rangle \\ &= \langle f(x,t), \mathcal{L}g(x,t)\rangle.\end{aligned}$$

Hence $\frac{\partial}{\partial t}g(x,t) = \mathcal{L}g(x,t)$.

To determine $\mathcal{L}$ and to find out which boundary conditions g has to satisfy, we expand the equation

$$\langle \mathcal{L}^* f(x,t), g(x,t)\rangle = \langle f(x,t), \mathcal{L}g(x,t)\rangle.$$

Using integration by parts, and substituting in (weaker) boundary condition (B.1) we have that

$$
\begin{aligned}
\langle \mathcal{L}^* f(x,t), g(x,t) \rangle &= \int_0^1 \frac{\partial}{\partial x} \left(\frac{1}{2} \frac{\partial}{\partial x} (b(x) f(x,t)) - a(x) f(x,t) \right) g(x,t) dx \\
&+ \frac{\partial}{\partial t} f(1,t) g(0,t) + \frac{\partial}{\partial t} f(1,t) g(1,t) \\
&= - \int_0^1 \left(\frac{1}{2} b(x) \frac{\partial}{\partial x} f(x,t) - a(x) f(x,t) \right) \frac{\partial}{\partial x} g(x,t) dx \\
&+ \left(\frac{1}{2} b(x) \frac{\partial}{\partial x} f(x,t) - a(x) f(x,t) \right) g(x,t) \Big|_{x=0}^1 \\
&- \left(\frac{1}{2} b(x) \frac{\partial}{\partial x} f(x,t) - a(x) f(x,t) \right) f(x,t) \Big|_{x=0}^1 \\
&= \int_0^1 \frac{1}{2} b(x) \frac{\partial}{\partial x} f(x,t) \frac{\partial}{\partial x} g(x,t) dx - \int_0^1 a(x) f(x,t) \frac{\partial}{\partial x} g(x,t) dx \\
&= \int_0^1 f(x,t) \left(\frac{1}{2} b(x) \frac{\partial^2}{\partial x^2} g(x,t) dx + a(x) \frac{\partial}{\partial x} g(x,t) \right) \\
&- f(x,t) \frac{1}{2} b(x) \frac{\partial}{\partial x} g(x,t) \Big|_{x=0}^1 \\
&= \langle f(x,t), \mathcal{L} g(x,t) \rangle - f(x,t) \frac{\partial}{\partial t} g(x,t) \Big|_{x=0,1} \\
&- f(x,t) \frac{1}{2} b(x) \frac{\partial}{\partial x} g(x,t) \Big|_{x=0}^1
\end{aligned}
$$

in order to obtain

$$
\mathcal{L} g(x,t) = \frac{1}{2} b(x) \frac{\partial^2}{\partial x^2} g(x,t) + a(x) \frac{\partial}{\partial x} g(x,t).
$$

we also have the boundary conditions

$$
\left[\frac{\partial}{\partial t} g(x,t) \right]_{t=0}^1 = \left[-f(x,t) \left(\frac{1}{2} \frac{\partial}{\partial x} (b(x) g(x,t)) \right) \right]_{x=0,1},
$$

which with the assumption of independence at $0, 1$ reduce to the boundary conditions

$$
\begin{aligned}
\frac{\partial}{\partial t} g(0,t) &= \frac{1}{2} \frac{\partial}{\partial x} (b(x) g(x,t)) \Big|_{x=0}, \\
\frac{\partial}{\partial t} g(1,t) &= - \frac{1}{2} \frac{\partial}{\partial x} (b(x) g(x,t)) \Big|_{x=1}.
\end{aligned}
$$

These boundary conditions appears to be satisfied when $b(x) = x(1-x)$ given that

$g(x,t)$ is smooth. Some work remains to be done in order to establish the regularity of $g(x,t)$ when density solution $f(x,t)$ are allowed to have point masses on the boundary. Currently this is only of theoretical interest and outside the scope of the thesis since we assume in application that $\beta_1, \beta_2 > 0$.

Appendix C

Derivations for Master Lemma 3.5.1

In order to find the explicit expressions for Master Lemma 3.5.1 (Chapter 3) we first derive some preliminary results.

Lemma C.0.1. *We establish (as expected) that the Chebyshev integral formula and Chebyshev differential operator cancels out:*

$$4(k+1)\int \sum_{\substack{r=0 \\ n-r \ odd}}^{k} T_r^*(x)dx = T_{k+1}^*(x).$$

Note that $\sum'$ *indicates taking only half of the first term when taking the sum.*

Proof. We consider the case when k is even,

$$\begin{aligned}
4(k+1)\int \sum_{\substack{r=0 \\ n-r \text{ even}}}^{k} T_r^*(x)dx =& 4(k+1)\int \left(\frac{1}{2}T_0^*(x) + T_2^*(x) + \cdots T_k^*(x)\right)dx \\
=& 4(k+1)\left(\frac{1}{2}\left(\frac{T_1^*(x)}{2}\right) + \frac{1}{4}\left(\frac{T_3^*(x)}{3} - \frac{T_1^*(x)}{1}\right)\right. \\
&+ \frac{1}{4}\left(\frac{T_5^*(x)}{5} - \frac{T_3^*(x)}{3}\right) \\
&+ \cdots + \frac{1}{4}\left(\frac{T_{k-1}(x)}{k-1} - \frac{T_{k-3}^*(x)}{k-3}\right) \\
&\left.+ \frac{1}{4}\left(\frac{T_{k+1}^*(x)}{k+1} - \frac{T_{k-1}^*(x)}{k-1}\right)\right) \\
=& 4(k+1)\left(\frac{T_1^*(x)}{1} - \frac{T_1^*(x)}{1}\right) + 4(k+1)\left(\frac{T_3(x)}{3} - \frac{T_3^*(x)}{3}\right) \\
&+ \cdots + 4(k+1)\left(\frac{T_{k-3}^*(x)}{k-3} - \frac{T_{k-3}^*(x)}{k-3}\right) + T_{k+1}^*(x) \\
=& T_{k+1}^*(x).
\end{aligned}$$

The same can be shown when k is odd. □

We now apply Lemma C.0.1 to the components of the double integrated forward PDE in terms of a Chebyshev basis function k to help simplify the derivation of the explicit expressions in Master Lemma 3.5.1.

Lemma C.0.2. *For any positive integer k we can write x and the Chebyshev differential formula in terms of a simple explicit expression when integrated as*

$$\int x\frac{\partial}{\partial x}T_k^*(x)dx = \frac{k}{4(k-1)}T_{k-1}^*(x) + \frac{1}{2}T_k^*(x) + \frac{k}{4(k+1)}T_{k+1}^*(x).$$

Proof. After reshuffling some of the terms this follows from applying Lemma C.0.1 to

each term separately as shown below

$$\begin{aligned}
\int x\frac{\partial}{\partial x}T_k^*(x)dx &= \int x\cdot 4k \sum_{\substack{r=0\\ k-r \text{ odd}}}^{k} T_r^*(x)dx \\
&= \int 4k\Bigg(\frac{1}{4}T_0^*(x)+\frac{1}{4}T_1^*(x) \\
&\quad + \sum_{\substack{r=1\\ k-r\text{odd}}}^{k-1}\left(\frac{1}{4}T_{r-1}^*(x)+\frac{1}{2}T_r^*(x)+\frac{1}{4}T_{r+1}^*(x)\right)\Bigg)dx \\
&= \int 4k\left(\sum_{\substack{r=0\\ k-r\text{odd}}}^{k-2}\frac{1}{4}T_r^*(x)+\sum_{\substack{r=0\\ k-r\text{odd}}}^{k-1}{}'\frac{1}{2}T_r^*(x)+\sum_{\substack{r=1\\ k-r\text{odd}}}^{k}\frac{1}{4}T_r^*(x)\right)dx \\
&= \frac{4(k-1)k}{4(k-1)}\int \sum_{\substack{r=0\\ k-r\text{odd}}}^{k-2} T_r^*(x)dx + 4k\int \sum_{\substack{r=0\\ k-r\text{odd}}}^{k-1}\frac{1}{2}T_r^*(x)dx \\
&\quad + \frac{4(k+1)k}{4(k+1)}\int \sum_{\substack{r=0\\ k-r\text{odd}}}^{k} T_r^*(x)dx \\
&= \frac{k}{4(k-1)}T_{k-1}^*(x)+\frac{1}{2}T_k^*(x)+\frac{k}{4(k+1)}T_{k+1}^*(x).
\end{aligned}$$

□

Lemma C.0.3. *For any positive integer $k > 1$ we can write x and the Chebyshev differential formula of $T_k^*(x)$ in terms of a simple explicit expression when integrated twice as*

$$\begin{aligned}
\int\int x\frac{\partial}{\partial x}T_k^*(x)dxdy = &-\frac{k}{16(k-1)(k-2)}T_{k-2}^* - \frac{1}{8(k-1)}T_{k-1}^* \\
&+ \frac{1}{8\left(k^2-1\right)}T_k^* + \frac{1}{8(k+1)}T_{k+1}^*\frac{n}{16(k+1)(k+2)}T_{k+2}^*.
\end{aligned}$$

Proof.

$$\begin{aligned}\int\int x\frac{\partial}{\partial x}T_k^* dxdy =& \int\left(\frac{k}{4(k-1)}T_{k-1}^* + \frac{1}{2}T_k^* + \frac{k}{4(k+1)}T_{k+1}^*\right)dx \\ =& \frac{k}{16(k-1)}\left(\frac{1}{k}T_k^* - \frac{1}{(k-2)}T_{k-2}^*\right) \\ &+ \frac{1}{8}\left(\frac{1}{(k+1)}T_{k+1}^* - \frac{1}{(k-1)}T_{k-1}^*\right) \\ &+ \frac{k}{16(k+1)}\left(\frac{1}{(k+2)}T_{k+2}^* - \frac{1}{k}T_k^*\right) \\ =& -\frac{k}{16(k-1)(k-2)}T_{k-2}^* - \frac{1}{8(k-1)}T_{k-1}^* \\ &+ \frac{1}{8\left(k^2-1\right)}T_k^* + \frac{1}{8(k+1)}T_{k+1}^* \frac{k}{16(k+1)(k+2)}T_{k+2}^*.\end{aligned}$$

□

Lemma C.0.4. *For any positive integer $k > 2$ we can write x and the second derivative of a Chebyshev polynomial $T_k^*(x)$ in terms of a simple explicit expression when integrated twice as*

$$\begin{aligned}\int\int x\frac{\partial^2}{\partial x^2}T_k^*(x)dxdy = \frac{k+1}{4(k-1)}T_{k-1}^*(x) \\ + \frac{3}{8}T_k^*(x) + \frac{k-1}{4(k+1)}T_{k+1}^*(x).\end{aligned}$$

Proof.

$$\begin{aligned}
\int\int x\frac{\partial^2}{\partial x^2}T_k^*(x)dxdy &= \int 4k \sum_{\substack{r=0\\ k-r \text{ odd}}}^{k-1} \int 4r \sum_{\substack{j=0\\ k-r \text{ odd}}}^{r-1}{}' \; x\cdot T_r^*(x)dxdy \\
&= \int 4n\left(\frac{k}{4(k-1)}T_{k-1}^*(x)+\frac{1}{2}T_k^*(x)+\frac{n}{4(k+1)}T_{k+1}^*(x)\right)dx \\
&= \int 4k\left(\left(\frac{1}{2}T_1^*+\frac{1}{8}T_2^*\right)+\left(\frac{3}{8}T_2^*+\frac{1}{2}T_3^*+\frac{3}{16}T_4^*\right)+\cdots\right. \\
&\quad +\left(\frac{k-3}{4(k-4)}T_{k-4}^*+\frac{1}{2}T_{k-3}^*+\frac{k-3}{4(k-2)}T_{k-2}^*\right) \\
&\quad \left.+\left(\frac{k-1}{4(k-2)}T_{k-2}^*+\frac{1}{2}T_{k-1}^*+\frac{k-1}{4k}T_k^*\right)\right)dx \\
&= \int 4k\left(\frac{1}{2}T_1^*+\frac{1}{2}T_2^*+\frac{1}{2}T_3^*+\cdots+\frac{1}{2}T_{k-1}^*+\frac{k-1}{4k}T_k^*\right)dx \\
&= \frac{k}{4}T_2^*+\frac{k}{2}\left(\frac{1}{3}T_3^*-T_1\right)+\frac{k}{2}\left(\frac{1}{4}T_4^*-\frac{1}{2}T_2^*\right)+\cdots \\
&\quad +\frac{k}{2}\left(\frac{1}{k-1}T_{k-1}^*-\frac{1}{k-3}T_{k-3}^*\right)+\frac{k}{2}\left(\frac{1}{k}T_k^*-\frac{1}{k-2}T_{k-2}^*\right) \\
&\quad +\frac{k-1}{4}\left(\frac{1}{k+1}T_{k+1}^*-\frac{1}{k-1}T_{k-1}^*\right) \\
&= \frac{k}{2}T_1^*+2k\left(\frac{1}{8}T_2^*-\frac{1}{8}T_2^*\right)+2k\left(\frac{1}{12}T_3^*-\frac{1}{12}T_3^*\right)+\cdots \\
&\quad +2k\left(\frac{1}{k-2}T_{k-2}^*-\frac{1}{k-2}T_{k-2}^*\right) \\
&\quad +\left(\frac{2k}{4(k-1)}T_{k-1}^*-\frac{1}{4}T_{k-1}^*\right)+\frac{1}{2}T_k^*+\frac{k-1}{4(k+1)}T_{k+1}^* \\
&= \frac{k+1}{4(k-1)}T_{k-1}^*(x)+\frac{3}{8}T_k^*(x)+\frac{k-1}{4(k+1)}T_{k+1}^*(x).
\end{aligned}$$

□

Lemma C.0.5. *For any positive integer $k > 2$ we can write x^2 and the derivative of a Chebyshev polynomial $T_k^*(x)$ in terms of a simple explicit expression when integrated twice.*

$$\begin{aligned}
\int x^2\frac{\partial}{\partial x}T_k^*(x)dxdy &= \frac{k}{16(k-2)}T_{k-2}^*(x)+\frac{k}{4(k-1)}T_{k-1}^*(x)+\frac{3}{8}T_k^*(x) \\
&\quad +\frac{k}{4(k+1)}T_{k+1}^*(x)+\frac{k}{16(k-1)}T_{k+2}^*(x).
\end{aligned}$$

Proof.

$$\begin{aligned}
\int x^2 \frac{\partial}{\partial x} T_k^*(x) dx =& \int 4k \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k-1}{}' \; x^2 \cdot T_r^*(x) dx \\
=& 4k \int \left(\frac{3}{8} T_0^* + \frac{1}{2} T_1^* + \frac{1}{8} T_2^* \right) + \left(\frac{1}{4} T_0^* + \frac{1}{16} T_1^* + \frac{1}{4} T_2^* + \frac{1}{16} T_3^* \right) \\
&+ \sum_{\substack{r=2 \\ k-r \text{ odd}}}^{k-1} \left(\frac{1}{16} T_{r-2}^* + \frac{1}{4} T_{r-1}^* + \frac{3}{8} T_r^* + \frac{1}{4} T_{r+1}^* + \frac{1}{16} T_{r+2}^* \right) dx \\
=& \int 4k \left(\sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k-3} \frac{1}{16} T_r^* + \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k-2} \frac{1}{4} \, T_r^* \right. \\
&\left. + \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k-1} \frac{3}{8} T_r^* + \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k} \frac{1}{4} T_r^* + \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k+1} \frac{1}{16} T_r^* \right) dx \\
=& \frac{k}{16(k-2)} T_{k-2}^* + \frac{k}{4(k-1)} T_{k-1}^* + \frac{3}{8} T_k^* \\
&+ \frac{k}{4(k+1)} T_{k+1}^*(x) + \frac{k}{16(k-1)} T_{k+2}^*.
\end{aligned}$$

□

Lemma C.0.6. *For any $k > 0$ we can write x^2 and the second derivative of a Chebyshev polynomial $T_k^*(x)$ in terms of a simple explicit expression when integrated twice.*

$$\begin{aligned}
\int \int x^2 \frac{\partial^2}{\partial x^2} T_k^*(x) dx dy =& 2k T_1^*(x) + \frac{k(k+1)}{16\left(k^2 - 3k + 2\right)} T_{k-2}^*(x) \\
&+ \frac{k+1}{4(k-1)} T_{k-1}^*(x) + \frac{5 - 3k^2}{8 - 8k^2} T_k^*(x) \\
&+ \frac{k-1}{4(k+1)} T_{k+1}^*(x) + \frac{(k-1)k}{16(k+1)(k+2)} T_{k+2}^*(x), \text{ for } k > 1.
\end{aligned}$$

Proof.

$$
\begin{aligned}
\int\int x^2 \frac{\partial^2}{\partial x^2} T_k^*(x)dxdy &= \int 4k \sum_{\substack{r=0 \\ k-r \text{ odd}}}^{k-1} \int 4r \sum_{\substack{j=0 \\ k-r \text{ odd}}}^{r-1}{}' \; x^2 \cdot T_r^*(x)dxdy \\
&= \int 4k \sum_{\substack{r=2 \\ k-r \text{ odd}}}^{k-1} \left(\frac{r}{16(r-2)} T_{r-2}^*(x) + \frac{r}{4(r-1)} T_{r-1}^*(x) \right. \\
&\quad \left. + \frac{3}{8} T_r^*(x) + \frac{r}{4(r+1)} T_{r+1}^*(x) + \frac{r}{16(r-1)} T_{r+2}^*(x) \right) dx \\
&= \int 4k \left(\left(\frac{1}{2} T_1^* + \frac{3}{8} T_2^* + \frac{1}{6} T_3^* + \frac{1}{32} T_4^* \right) \right. \\
&\quad + \left(\frac{1}{8} T_2^* + \frac{1}{3} T_3^* + \frac{3}{8} T_4^* + \frac{1}{5} T_5^* + \frac{1}{24} T_6^* \right) \\
&\quad + \left(\frac{6}{64} T_4^* + \frac{6}{20} T_5^* + \frac{3}{8} T_6^* + \frac{6}{28} T_7^* + \frac{6}{128} T_8^* \right) \\
&\quad \vdots \\
&\quad + \left(\frac{k-3}{16(k-5)} T_{k-5}^* + \frac{k-3}{4(k-4)} T_{k-4}^* \right. \\
&\quad \left. + \frac{3}{8} T_{k-3}^* + \frac{(k-3)}{4(k-2)} T_{k-2}^* + \frac{(k-3)}{16(k-1)} T_{k-1}^* \right) \\
&\quad + \left(\frac{k-1}{16(k-3)} T_{k-3}^* + \frac{k-1}{4(k-2)} T_{k-2}^* \right. \\
&\quad \left. \left. + \frac{3}{8} T_{k-1}^* + \frac{(k-1)}{4(k)} T_k^* + \frac{(k-1)}{16(k+1)} T_{k+1}^* \right) \right) \\
&= \int 4k \left(\frac{1}{2} T_1^* + \frac{1}{2} T_2^* + \frac{1}{2} T_3^* + \frac{1}{2} T_4^* + \cdots + \frac{1}{2} T_{k-2}^* \right. \\
&\quad + \frac{9-7k}{16-16k} T_{k-1}^* + \frac{k-1}{4(k+1)} T_{k+1}^*(x) \\
&\quad \left. + \frac{(k-1)k}{16(k+1)(k+2)} T_{k+2}^*(x) \right) dx \\
&= 2kT_1^* + \frac{k(k+1)}{16\left(k^2-3k+2\right)} T_{k-2}^* + \frac{k+1}{4(k-1)} T_{k-1}^* \\
&\quad + \frac{5-3k^2}{8-8k^2} T_k^* + \frac{k-1}{4(k+1)} T_{k+1}^* + \frac{(k-1)k}{16(k+1)(k+2)} T_{k+2}^*.
\end{aligned}
$$

□

Appendix D

Probability density functions

We give the formulas for the PDFs used in Chapter 5 to specify priors on the model parameters (see Section 5.6.1 for more detail on the relevant model parameters).

Chi-squared distribution

Suppose that ν is a positive integer that specifies the degrees of freedom then

$$f(x, \nu) = \begin{cases} \frac{x^{\nu/2-1}e^{-x/2}}{2^{\nu/2}\Gamma(\nu/2)}, & x > 0, \\ 0, & \text{otherwise.} \end{cases}.$$

See Forbes *et al.* (2011, pg. 69).

Gamma distribution

Suppose that $\alpha > 0$ (specifies shape) and $\beta > 0$ (specifies rate) then

$$f(x, \alpha, \beta) = \frac{\beta^{\alpha} x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}, \qquad \text{for } x > 0.$$

See Forbes *et al.* (2011, pg. 109).

Inverse-wishart distribution

Suppose Ψ is a $K \times K$ scale matrix and ν the degrees of freedom then

$$f(\mathbf{x}, \Psi, \nu) = \frac{|\Psi|^{\nu/2}}{2^{\nu K/2}\Gamma_K(\frac{\nu}{2})|\mathbf{x}|^{-(\nu+K+1)/2}} e^{-\frac{1}{2}\text{tr}(\Psi \mathbf{x}^{-1})},$$

where Γ_K is a multivariate gamma function

$$\Gamma_K\left(\frac{\nu}{2}\right) = \pi^{(\frac{\nu}{2})(\frac{\nu}{2}-1)/4}\prod_{j=1}^{K}\Gamma\left(\left(\frac{\nu}{2}\right)+(1-j)/2\right).$$

and $|.|$ is the matrix determinant. See Forbes *et al.* (2011, pg. 209).

Symmetric Dirichlet distribution

Suppose that $\alpha > 0$ then

$$f(\gamma_1,\ldots,\gamma_{K^2},\alpha) = \frac{\Gamma(\alpha K^2)}{\alpha^{K^2}}\prod_{i=1}^{K^2}\gamma_i^{\alpha-1},$$

we note that probability mass is sparsely distributed among $\gamma_1,\ldots,\gamma_{K^2}$ if $\alpha < 1$. See Forbes *et al.* (2011, pg. 78).

Appendix E

Tabulated posterior statistics

Chapter 3

Table E.1: Combined SNAPPER parameter summary after 1,000,000 MCMC iterations for soybean dataset

Snapper	**mean**	**variance**	**HPD$_{[05,95]}$**	**ACT**	**ESS**
posterior	-6688959	14.7676	[-6688967, -6688952]	8907.766	266.7335
theta0	0.0103	6.9701E-09	[0.0101, 0.0104]	44789.7957	53.0478
theta1	0.0171	2.143E-08	[0.0169, 0.0174]	78994.8943	30.0779
theta2	0.0095108	5.5572E-09	[9.37E-3, 9.64E-3]	11003.7147	215.9271
theta3	0.0096895	5.3618E-09	[9.54-3, 9.81E-3]	8753.8683	271.4229
theta4	0.0293	3.434E-08	[0.0289, 0.0297]	1104.855	2150.5085
theta5	0.0167	1.2541E-08	[0.0165, 0.0169]	25804.9788	92.0753
theta6	0.0501	1.5005E-07	[0.0494, 0.0508]	55201.1873	43.0426
theta7	0.0301	3.4758E-08	[0.0298, 0.0305]	1099.2779	2161.4188
theta8	0.0194	2.071E-08	[0.0191, 0.0197]	1000	2376
theta9	0.0115	6.1131E-09	[0.0113, 0.0116]	29679.2095	80.056
theta10	0.0748	3.0959E-06	[0.0713, 0.078]	23976.3835	99.0975
theta11	0.0991	3.2988E-06	[0.0957, 0.1024]	59333.0205	40.0452
theta12	0.0799	2.1569E-06	[0.0771, 0.0827]	38860.524	61.1417
theta13	0.1924	6.2945E-06	[0.1877, 0.1977]	70510.2825	33.6972
theta14	0.2544	1.668E-05	[0.2457, 0.2619]	19664.1926	120.8288
theta15	0.5017	3.7525E-06	[0.4981, 0.5053]	33043.8999	71.9043
theta16	0.0091159	3.2102E-06	[6.0399E-3, 0.0128]	20602.006	115.3286
theta17	0.0107013	1.7778E-06	[5.1511E-3, 0.0102]	12049.29	197.19
theta18	0.0123	1.6687E-05	[5.5037E-3, 0.0205]	22763.4335	104.3779
tree.height	0.3902	1.8269E-05	[0.3873, 0. 3961]	24967.0853	95.1653
Likelihood	-6686132	244.8637	[-6686160, -6686102]	86900.8557	27.3415
prior	-2826.6072	249.9542	[-2855, -2794]	93838.3491	25.3201

Table E.2: Combined SNAPP parameter summary after 1,000,000 MCMC iterations for soybean dataset

Snapp	**mean**	**variance**	**HPD$_{[05,95]}$**	**ACT**	**ESS**
posterior	-6691316	15.911	[-6691324, -6691309]	8233.9744	287.8607
theta0	0.0111	5.9541E-09	[0.0109, 0.0112]	19051.0563	119.5265
theta1	0.0191	1.4173E-08	[0.0189, 0.0193]	44905.8468	58.284
theta2	0.0099244	6.8368E-09	[9.7664E-3, 0.0101]	2445.2984	305.45
theta3	0.0101	5.8762E-09	[0.01, 0.0103]	2398.1736	216.6085
theta4	0.0293	3.9721E-08	[0.029, 0.0297]	1826.4339	2233.3791
theta5	0.0178	1.0608E-08	[0.0176, 0.018]	2768.2884	500.3791
theta6	0.0562	7.4177E-08	[0.0556, 0.0566]	11585.9093	132.108
theta7	0.0301	2.691E-08	[0.0298, 0.0305]	1497.2333	2248.45832
theta8	0.0194	2.1582E-08	[0.0191, 0.0197]	1608.6939	2231.2435
theta9	0.0122	6.7513E-09	[0.012, 0.0123]	11598.0129	232.0745
theta10	0.0791	1.705E-06	[0.0766, 0.0815]	3476.1242	207.0157
theta11	0.1031	1.2658E-06	[0.1012, 0.1053]	14850.7685	125.0492
theta12	0.0867	1.4519E-06	[0.0846, 0.0891]	12046.2026	330.8811
theta13	0.194	3.8296E-06	[0.1906, 0.1979]	37167.9581	110.0086
theta14	0.2456	9.2542E-06	[0.2404, 0.2518]	4309.2232	186.3265
theta15	0.464	2.7819E-06	[0.4613, 0.4678]	15690.0573	223.7093
theta16	0.0184	7.8788E-06	[0.0127, 0.0233]	18680.3197	119.914
theta17	0.0145	3.9414E-06	[0.0105, 0.0178]	19410.3875	119.165
theta18	0.0176	1.4512E-05	[0.0101, 0.0245]	14593.2873	25.4912
tree.height	0.3912	4.0822E-06	[0.3878, 0.3951]	16143.5651	223.0432
Likelihood	-6686791	594.6324	[-6686834, -6686741]	59846.9891	56.2159
prior	-4524.8505	612.8955	[-4578, -4484]	63835.4742	55.8275

Table E.3: SNAPPER parameter summary after 2,000,000 MCMC iterations for freshwater turtle dataset

Snapper	**mean**	**variance**	**HPD$_{[05,95]}$**	**ACT**	**ESS**
posterior	-32288	58.555	[-32289,-32286]	6921.9588	237.4934
theta0	0.0892	1.09E-05	[0.0829,0.0956]	4559.0819	360.5814
theta1	0.0326	2.03E-06	[0.0299,0.0354]	3334.7123	492.972
theta2	0.0326	2.31E-06	[0.0297,0.0355]	4882.2185	336.7158
theta3	0.0315	2.28E-06	[0.0288,0.0347]	4594.6906	357.7868
theta4	0.089	1.16E-05	[0.0823,0.0955]	5834.7779	281.745
theta5	0.0209	7.38E-07	[0.0195,0.0228]	5032.2875	326.6746
theta6	0.0209	7.52E-07	[0.0193,0.0227]	5336.7619	308.037
theta7	0.0211	7.42E-07	[0.0195,0.0227]	5739.448	286.4248
theta8	0.0568	5.41E-06	[0.0525,0.0613]	7592.2756	216.5254
theta9	0.0576	5.36E-06	[0.0533,0.0622]	5374.5587	305.8706
theta10	0.0576	5.40E-06	[0.0535,0.0621]	5700.7353	288.3698
theta11	0.0398	3.51E-06	[0.0362,0.0437]	4824.2314	340.763
theta12	0.0475	2.82E-06	[0.0439,0.0506]	5591.2287	294.0176
theta13	0.0474	3.08E-06	[0.0441,0.0508]	7868.6887	208.9192
theta14	0.0473	3.13E-06	[0.0441,0.0509]	6942.6134	236.787
theta15	0.0474	2.99E-06	[0.0441,0.0507]	7085.1136	232.0246
theta16	0.0473	2.84E-06	[0.0442,0.0503]	4927.4417	333.6254
theta17	0.0563	5.91E-06	[0.0512,0.0608]	6188.3147	265.649

Continued on next page

Table E.3 – *Continued from previous page*

Snapper	**mean**	**variance**	**HPD**$_{[05,95]}$	**ACT**	**ESS**
theta18	0.0254	1.90E-06	[0.0229,0.0283]	4461.407	368.4756
theta19	0.021	1.58E-06	[0.0186,0.0233]	3920.8397	419.2776
theta20	0.0351	2.47E-06	[0.032,0.0381]	5149.6604	319.2288
theta21	0.035	2.45E-06	[0.032,0.0382]	5103.4682	322.1182
theta22	0.0312	1.98E-06	[0.0286, 0.034]	5843.4713	281.326
theta23	0.0275	1.68E-06	[0.0249,0.0298]	4633.1697	354.8154
theta24	0.0137	5.95E-07	[0.0121,0.0151]	6877.4641	239.03
theta25	0.0137	6.94E-07	[0.0121,0.0152]	7605.5597	216.1472
theta26	0.0105	3.66E-07	[9.3707E-3,0.0116]	5984.3981	274.701
theta27	0.035	1.34E-06	[0.0328,0.0371]	4350.192	377.896
theta28	0.0324	1.52E-06	[0.0301,0.0349]	6937.3698	236.9658
theta29	0.0324	1.51E-06	[0.03,0.0347]	6434.1179	255.5004
theta30	0.033	1.42E-06	[0.0305,0.0352]	5917.0715	277.8266
theta31	0.033	1.34E-06	[0.0304,0.035]	5047.0492	325.719
theta32	0.0346	1.40E-06	[0.0325,0.0371]	5272.0715	311.8168
theta33	0.0345	1.55E-06	[0.0321,0.0367]	4934.2925	333.1622
theta34	0.0346	1.52E-06	[0.0322,0.037]	5797.7463	283.5446
theta35	0.0345	1.39E-06	[0.0319,0.0366]	5713.2156	287.7398
theta36	0.0343	1.49E-06	[0.0321,0.0367]	5738.9424	286.45
theta37	0.032	1.73E-06	[0.0296,0.0346]	5566.4597	295.326
theta38	8.24E-03	1.83E-07	[7.3532E-3,9.0356E-3]	4660.0198	352.771
theta39	0.0285	2.06E-06	[0.0254,0.031]	2754.3786	596.8388
theta40	0.045	3.94E-06	[0.0416,0.0493]	2735.1968	601.0244
theta41	0.0891	1.03E-05	[0.0823,0.0946]	4363.7001	376.7262
theta42	0.0328	1.74E-06	[0.0302,0.0352]	4046.0439	406.303
theta43	0.0344	1.63E-06	[0.0321,0.037]	4026.7212	408.2528
theta44	0.0553	2.70E-06	[0.0522,0.0585]	7152.4991	229.8386
theta45	0.0209	7.09E-07	[0.0196,0.023]	4865.2479	337.8902
theta46	0.021	6.24E-07	[0.0195,0.0226]	4733.2039	347.3166
theta47	0.0518	4.84E-06	[0.0481,0.0555]	26723.5617	61.5158
theta48	0.0575	4.88E-06	[0.0538,0.0621]	5530.8802	297.2258
theta49	0.0573	4.28E-06	[0.0535,0.0612]	6291.947	261.2736
theta50	0.0567	3.64E-06	[0.0531,0.0607]	7435.9237	221.0782
theta51	0.0475	2.81E-06	[0.0441,0.0506]	7190.8831	228.6116
theta52	0.0474	2.92E-06	[0.0442,0.0506]	6880.0099	238.9416
theta53	0.0473	2.65E-06	[0.0444,0.0507]	5266.9057	312.1226
theta54	0.0473	2.64E-06	[0.044,0.0502]	5930.9334	277.1772
theta55	0.0452	2.35E-06	[0.0423,0.0481]	6519.7251	252.1456
theta56	0.0531	2.40E-06	[0.0502,0.0562]	10545.795	155.884
theta57	0.0246	1.12E-06	[0.0225,0.0265]	6463.5958	254.3352
theta58	0.0351	2.26E-06	[0.0324,0.0382]	4395.8089	373.9744
theta59	0.0348	1.46E-06	[0.0326,0.0372]	12531.3457	131.1846
theta60	0.0387	1.12E-06	[0.0366,0.0405]	16803.2068	97.8338
theta61	0.0276	1.16E-06	[0.0254,0.0294]	11382.032	144.4312
theta62	0.0136	5.75E-07	[0.0122,0.015]	7378.0828	222.8112
theta63	0.0132	3.93E-07	[0.012,0.0144]	9066.0779	181.3264
theta64	0.025	7.92E-07	[0.0232,0.0266]	13972.7813	117.6516
theta65	0.0492	2.12E-06	[0.0465,0.052]	24561.4047	66.931
theta66	0.0345	1.30E-06	[0.0324,0.0367]	4752.1445	345.9322
theta67	0.0344	1.22E-06	[0.0323,0.0366]	4528.4154	363.0232
theta68	0.0345	1.28E-06	[0.0324,0.0368]	5549.8598	296.2092
theta69	0.0344	1.21E-06	[0.0323,0.0364]	4893.2052	335.9598

Continued on next page

Table E.3 – *Continued from previous page*

Snapper	mean	variance	$HPD_{[05,95]}$	ACT	ESS
theta70	0.0343	1.13E-06	[0.0323,0.0363]	4619.4067	355.8726
theta71	0.0324	1.44E-06	[0.0301,0.0346]	6038.6503	272.233
theta72	0.033	1.33E-06	[0.0304,0.035]	5481.3538	299.9114
theta73	0.0329	1.25E-06	[0.0305,0.0349]	5176.6725	117.563
theta74	0.0336	1.09E-06	[0.0317,0.0357]	4940.5417	332.7408
theta75	0.0334	1.06E-06	[0.0314,0.0354]	5603.2564	293.3866
theta76	0.037	3.01E-06	[0.0336,0.0401]	22711.2203	72.3836
theta77	0.0492	4.41E-06	[0.0456,0.0533]	33200.5358	49.5148
theta78	0.06	5.72E-06	[0.0558,0.064]	34594.3404	47.52
theta79	0.0383	9.33E-06	[0.0328,0.0438]	25502.3797	64.4614
theta80	0.0531	3.43E-06	[0.0495,0.0565]	33045.9805	49.7464
tree.height	0.1982	3.96E-05	[0.1863,0.2107]	1811.7552	907.3632
tree.length	0.754	2.41E-04	[0.7242,0.7845]	2135.0573	769.9652

Chapter 4

Table E.4: Parameter summary of TTR model for a single species after 300,000 MCMC iterations

single	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
posterior	49.6074	166.6980	42.1588	58.8689	344.1805	871.6357
beta1	0.1702	0.0145	0.0023	0.4005	456.3294	657.4199
beta2	0.4760	0.0440	0.0908	0.8552	395.6682	758.2111
beta3	0.8011	0.1833	0.0096	1.5524	423.8994	707.7152
beta4	1.4893	0.1188	0.8746	1.9988	430.9674	696.1084
beta5	0.3307	0.0567	0.0050	0.7679	461.0742	650.6545
beta6	0.6345	0.0570	0.2056	0.9984	451.8167	663.9861
beta7	0.1660	0.0042	0.0394	0.2570	546.4689	548.9791
beta8	0.7513	0.0309	0.4136	0.9952	514.0990	583.5452
beta9	0.2874	0.0193	0.0535	0.5047	545.5214	549.9326
beta10	0.9056	0.0097	0.6762	0.9986	531.7911	564.1313
beta11	0.0162	0.0001	0.0004	0.0348	503.3328	596.0272
beta12	0.0866	0.0027	0.0162	0.1489	531.1234	564.8405
beta13	0.8877	0.0081	0.8055	0.9774	541.1866	554.3374
beta14	0.9574	0.0016	0.9008	0.9996	484.4382	619.2741
beta15	1.0214	0.1745	0.1944	1.8037	421.8319	711.1837
beta16	1.6731	0.0537	1.2077	1.9987	427.5742	701.6326
beta17	0.2579	0.0279	0.0000	0.5252	478.4377	627.0409
beta18	0.6251	0.0538	0.2116	0.9997	457.4488	655.8111
beta19	0.8081	0.1985	0.0145	1.5922	408.0933	735.1260
beta20	1.5309	0.1059	0.9724	1.9997	435.7777	688.4244
beta21	0.4716	0.0936	0.0048	0.9549	433.8626	691.4632
beta22	0.4865	0.0770	0.0096	0.9418	423.1639	708.9451
beta23	0.5046	0.0920	0.0071	0.9515	428.5443	700.0444
beta24	0.4964	0.0844	0.0555	0.9987	422.6654	709.7812
beta25	0.2519	0.0210	0.0009	0.5174	495.6380	605.2805
beta26	0.5430	0.0416	0.1745	0.9275	420.9418	712.6875
beta27	1.0566	0.1825	0.2741	1.8902	457.7861	655.3279

Continued on next page

Table E.4 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
beta28	1.6085	0.0765	1.0663	1.9970	432.0061	694.4347
beta29	1.2927	0.2305	1.0007	2.3131	495.2772	605.7214

Table E.5: Parameter summary of TTR model for two species with competition after 500,000 MCMC iterations

comp	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
posterior	141.0903	608.1420	56.1556	207.5037	892.3910	560.2925
beta1	0.1465	0.0091	0.0020	0.4025	770.2324	649.1547
beta2	0.4468	0.0353	0.1435	0.9209	752.2556	664.6677
beta3	0.7418	0.1769	0.0307	1.7177	746.8117	669.5128
beta4	1.4456	0.1463	0.8305	1.9957	794.7439	629.1335
beta5	0.4303	0.0864	0.0077	0.8117	842.2253	593.6654
beta6	0.6927	0.0619	0.2341	0.9993	816.0954	612.6735
beta7	0.2857	0.0026	0.1735	0.3489	909.1116	549.9875
beta8	0.5259	0.0492	0.1801	0.7784	894.0852	559.2308
beta9	0.2715	0.0050	0.0830	0.4160	880.9446	567.5726
beta10	0.4195	0.0070	0.5360	0.9996	842.4466	593.5095
beta11	0.0127	0.0001	0.0000	0.0290	777.4907	643.0945
beta12	0.0330	0.0002	0.0040	0.0839	817.9610	611.2761
beta13	0.7557	0.0275	0.8990	0.9783	869.1291	575.2885
beta14	0.9467	0.0062	0.9512	0.9998	879.8978	568.2478
beta15	0.7711	0.2158	0.1546	1.8208	892.5610	560.1858
beta16	1.7327	0.0344	1.1839	1.9962	825.5083	605.6874
beta17	0.1945	0.0183	0.0016	0.5032	800.0680	624.9469
beta18	0.5167	0.0446	0.2361	0.9936	745.9546	670.2821
beta19	0.7521	0.2249	0.0189	1.5804	815.0233	613.4794
beta20	1.3683	0.2120	0.6791	1.9918	839.3376	595.7078
beta21	0.5595	0.0763	0.0040	0.9410	725.6455	689.0417
beta22	0.4893	0.0830	0.0618	0.9867	759.2191	658.5714
beta23	0.5018	0.0805	0.0064	0.9506	749.9133	666.7437
beta24	0.4894	0.0841	0.0011	0.9294	758.1347	659.5134
beta25	0.1670	0.0178	0.0021	0.4758	833.3337	599.9997
beta26	0.5335	0.0613	0.1662	0.9996	790.7394	632.3196
beta27	0.5077	0.0386	0.0007	1.5710	864.0818	578.6489
beta28	0.7994	0.0811	0.5322	1.9934	876.7942	570.2593
beta29	1.4240	0.3132	1.0007	2.9855	827.8002	604.0105
beta30	0.1688	0.0143	0.0012	0.3188	749.2629	667.3225
beta31	0.4855	0.0446	0.0751	0.7796	710.4282	703.8009
beta32	0.8881	0.2319	0.0007	1.5578	732.4708	682.6210
beta33	1.5363	0.1260	0.6956	1.9903	749.2727	667.3138
beta34	0.3601	0.0622	0.0284	0.9975	763.1591	655.1714
beta35	0.7088	0.0552	0.2364	0.9999	770.5382	648.8971
beta36	0.2360	0.0021	0.2226	0.3674	901.7721	554.4638
beta37	0.3952	0.0302	0.2703	0.9316	873.9240	572.1321
beta38	0.2894	0.0094	0.0857	0.3587	895.8715	558.1158
beta39	0.7930	0.0241	0.2452	0.5962	900.6696	555.1425
beta40	0.0122	0.0001	0.0001	0.0287	736.9036	678.5148
beta41	0.0416	0.0005	0.0108	0.0613	720.0299	694.4156
beta42	0.9423	0.0062	0.3983	0.9306	898.1089	556.7253
beta43	0.9752	0.0030	0.8713	0.9997	883.5599	565.8926

Continued on next page

Table E.5 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
beta44	0.9693	0.1942	0.2279	1.6354	743.6406	672.3678
beta45	1.6597	0.0589	1.4049	1.9957	770.7162	648.7472
beta46	0.2433	0.0247	0.0009	0.4200	750.5740	666.1568
beta47	0.5744	0.0424	0.1173	0.8979	713.9894	700.2905
beta48	0.7435	0.2022	0.0295	1.6385	727.6281	687.1643
beta49	1.4157	0.1526	0.5682	1.9918	759.3785	658.4331
beta50	0.4784	0.0881	0.0859	0.9902	713.8145	700.4621
beta51	0.5170	0.0835	0.0113	0.9477	711.5991	702.6428
beta52	0.4780	0.0810	0.0383	0.9832	683.2464	731.8004
beta53	0.4977	0.0874	0.0390	0.9796	742.5400	673.3644
beta54	0.2113	0.0219	0.0013	0.3950	757.5420	660.0294
beta55	0.5655	0.0527	0.1176	0.9751	743.1262	672.8332
beta56	0.7476	0.2242	0.0367	0.6836	824.6479	606.3194
beta57	1.3904	0.2133	0.5711	1.5184	841.9889	593.8321
beta58	1.5556	0.6662	1.0006	2.5040	845.7424	591.1965

Table E.6: Parameter summary of TTR model on three taxa species tree after 1,000,000 MCMC iterations

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
posterior	2500.90	149.86	2491.30	2509.40	936.57	1067.73
beta1	0.3048	0.0492	0.0049	0.7231	980.6079	1019.7756
beta2	0.5994	0.0646	0.2166	0.9692	992.8288	1007.2230
beta3	0.6071	0.2300	0.0280	1.5015	988.5247	1011.6085
beta4	1.3527	0.2123	0.4092	1.9948	985.2216	1015.0001
beta5	0.1457	0.0184	0.0007	0.5292	982.8947	1017.4030
beta6	0.5221	0.0582	0.1504	0.9063	981.7758	1018.5625
beta7	0.2615	0.0421	0.0074	0.6827	984.7072	1015.5303
beta8	0.5891	0.0740	0.1638	0.9940	987.9126	1012.2353
beta9	0.2688	0.0442	0.0001	0.6363	995.3876	1004.6338
beta10	0.6131	0.0414	0.2072	0.9338	982.9192	1017.3776
beta11	0.3175	0.0601	0.0025	0.7586	984.5594	1015.6828
beta12	0.5885	0.0699	0.1271	0.9947	994.4780	1005.5527
beta13	0.2697	0.0406	0.0051	0.6660	982.3238	1017.9943
beta14	0.6510	0.0429	0.3062	0.9937	983.2497	1017.0357
beta15	0.8113	0.1759	0.0860	1.5300	987.8582	1012.2910
beta16	1.3494	0.1828	0.5417	1.9907	986.3953	1013.7923
beta17	0.3063	0.0611	0.0045	0.7542	992.6384	1007.4162
beta18	0.6716	0.0565	0.2681	0.9970	993.2694	1006.7762
beta19	0.5074	0.1805	0.0039	1.4019	993.0231	1007.0259
beta20	1.2766	0.2034	0.4990	1.9854	986.8070	1013.3694
beta21	0.3388	0.0803	0.0119	0.9462	987.8832	1012.2654
beta22	0.4568	0.0747	0.0354	0.9032	982.6493	1017.6571
beta23	0.5112	0.1100	0.0363	0.9989	994.0816	1005.9536
beta24	0.5137	0.0569	0.0706	0.9074	986.0864	1014.1099
beta25	0.2660	0.0444	0.0322	0.7418	982.2770	1018.0428
beta26	0.6289	0.0578	0.1922	0.9535	986.0921	1014.1041
beta27	0.3034	0.0995	0.0041	0.8458	987.1896	1012.9766
beta28	1.2299	0.2914	0.1821	1.9689	988.3281	1011.8097
beta29	1.4260	0.0879	1.0322	1.8594	989.9705	1010.1311
beta30	0.2914	0.0472	0.0010	0.7021	991.2171	1008.8607

Continued on next page

Table E.6 – *Continued from previous page*

tree	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
beta31	0.5960	0.0747	0.1036	0.9904	986.1185	1014.0769
beta32	0.7744	0.1960	0.1523	1.5028	986.7613	1013.4163
beta33	1.3425	0.1816	0.4914	1.9556	988.6037	1011.5277
beta34	0.2812	0.0520	0.0105	0.7925	988.6970	1011.4322
beta35	0.6211	0.0538	0.1910	0.9930	988.3604	1011.7767
beta36	0.4453	0.0814	0.0228	0.8830	987.1365	1013.0311
beta37	0.8477	0.0282	0.3776	0.9971	992.3238	1007.7356
beta38	0.3302	0.0577	0.0013	0.7702	987.3623	1012.7995
beta39	0.6914	0.0523	0.1405	0.9998	981.4413	1018.9096
beta40	0.2955	0.0480	0.0027	0.8047	983.0935	1017.1972
beta41	0.5905	0.0647	0.1081	0.9549	992.1408	1007.9215
beta42	0.3311	0.0696	0.0088	0.8321	987.4333	1012.7266
beta43	0.7123	0.0528	0.2747	0.9901	987.6545	1012.4998
beta44	0.6844	0.1746	0.0640	1.4485	980.9135	1019.4579
beta45	1.3407	0.1766	0.6603	1.9691	989.0320	1011.0896
beta46	0.2911	0.0485	0.0015	0.6445	991.6081	1008.4629
beta47	0.6269	0.0449	0.1409	0.9573	976.1905	1024.3902
beta48	0.8603	0.2776	0.0508	1.7786	990.2441	1009.8520
beta49	1.6974	0.0926	0.9904	1.9821	977.0383	1023.5013
beta50	0.4299	0.0552	0.0062	0.8101	984.5476	1015.6949
beta51	0.3686	0.0924	0.0297	0.9915	987.5253	1012.6323
beta52	0.5024	0.0876	0.0515	0.9727	985.7229	1014.4839
beta53	0.5629	0.0857	0.0129	0.9882	988.9488	1011.1747
beta54	0.3341	0.0546	0.0320	0.7147	987.6736	1012.4802
beta55	0.6209	0.0476	0.2657	0.9926	989.0622	1011.0588
beta56	0.8346	0.2619	0.0009	1.6658	991.0892	1008.9909
beta57	1.4057	0.1727	0.6129	1.9635	991.2578	1008.8193
beta58	1.4481	0.0629	1.0056	1.8715	982.6410	1017.6657
beta59	0.3757	0.0696	0.0050	0.8009	993.8804	1006.1573
beta60	0.6742	0.0562	0.2124	0.9488	992.2670	1007.7933
beta61	0.6257	0.2166	0.0236	1.4081	991.0687	1009.0118
beta62	1.2233	0.1421	0.3792	1.8330	978.4920	1021.9808
beta63	0.3685	0.0431	0.0031	0.7922	980.4149	1019.9763
beta64	0.6772	0.0364	0.3485	0.9875	986.0020	1014.1967
beta65	0.3702	0.0475	0.0118	0.7360	991.6541	1008.4161
beta66	0.6534	0.0458	0.2909	0.9772	988.0871	1012.0565
beta67	0.4090	0.0563	0.0367	0.7809	985.1902	1015.0324
beta68	0.7060	0.0516	0.2530	0.9866	993.1495	1006.8978
beta69	0.3751	0.0448	0.0381	0.7773	987.8614	1012.2878
beta70	0.7010	0.0385	0.2804	0.9648	988.6662	1011.4637
beta71	0.3400	0.0445	0.0464	0.7676	986.1200	1014.0754
beta72	0.7297	0.0321	0.4086	0.9990	985.0839	1015.1420
beta73	0.5898	0.1320	0.0079	1.2927	983.3208	1016.9621
beta74	1.2610	0.1790	0.6806	1.9871	986.4649	1013.7208
beta75	0.4008	0.0641	0.0880	0.8047	986.0170	1014.1813
beta76	0.7080	0.0609	0.1901	0.9983	991.7088	1008.3605
beta77	0.6966	0.2418	0.0275	1.4618	982.1876	1018.1354
beta78	1.3524	0.2189	0.4522	1.9840	986.7595	1013.4182
beta79	0.4264	0.0837	0.0088	0.9169	983.3208	1016.9621
beta80	0.4341	0.0803	0.0353	0.9472	980.5256	1019.8612
beta81	0.7258	0.0832	0.0582	0.9882	992.3949	1007.6634
beta82	0.5748	0.0595	0.1107	0.9740	979.8846	1020.5283
beta83	0.3231	0.0458	0.0052	0.6702	985.1106	1015.1144
beta84	0.6702	0.0437	0.2520	0.9808	985.7026	1014.5048

Continued on next page

Table E.6 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
beta85	0.5171	0.1616	0.0413	1.3794	991.7270	1008.3420
beta86	1.2672	0.2688	0.4656	1.9907	986.9791	1013.1927
beta87	1.4698	0.1213	1.0036	1.9358	994.5581	1005.4717
beta88	0.0806	0.0040	0.0005	0.2105	989.7302	1010.3764
beta89	0.3227	0.0287	0.0582	0.6550	995.6742	1004.3446
beta90	1.2974	0.0225	1.0303	1.5339	990.7180	1009.3690
beta91	1.6518	0.0271	1.4202	1.9374	995.8637	1004.1535
beta92	0.2872	0.0249	0.0132	0.4958	997.7722	1002.2328
beta93	0.4936	0.0256	0.0731	0.6931	997.4037	1002.6031
beta94	0.4063	0.0184	0.0350	0.5398	996.4981	1003.5142
beta95	0.6157	0.0140	0.4641	0.8373	994.5687	1005.4610
beta96	0.5506	0.0333	0.1827	0.8059	996.5438	1003.4682
beta97	0.7694	0.0200	0.4713	0.9969	996.9688	1003.0404
beta98	0.4233	0.0388	0.1369	0.8261	997.9792	1002.0249
beta99	0.6623	0.0435	0.2764	0.9818	999.3610	1000.6394
beta100	0.4181	0.0223	0.1497	0.6124	995.3359	1004.6860
beta101	0.7014	0.0382	0.3303	0.9750	997.8763	1002.1282
beta102	0.5014	0.0375	0.2081	0.8318	994.5458	1005.4841
beta103	1.3678	0.0358	1.0252	1.7524	995.0342	1004.9906
beta104	0.3479	0.0169	0.1242	0.5505	987.8092	1012.3413
beta105	0.7299	0.0137	0.4838	0.9174	990.4463	1009.6459
beta106	0.3982	0.0346	0.0326	0.6554	996.7403	1003.2704
beta107	1.3819	0.0438	1.0283	1.7307	997.0144	1002.9945
beta108	0.4248	0.0343	0.1097	0.7366	995.9585	1004.0579
beta109	0.3403	0.0354	0.0176	0.6025	995.0609	1004.9636
beta110	0.7503	0.0192	0.5734	0.9955	992.9287	1007.1217
beta111	0.8372	0.0079	0.6524	0.9833	989.6621	1010.4459
beta112	0.3764	0.0080	0.2175	0.5179	994.3221	1005.7103
beta113	0.4984	0.0135	0.3322	0.7802	994.4798	1005.5508
beta114	0.6166	0.0372	0.3066	0.8840	994.8992	1005.1270
beta115	1.7020	0.0130	1.5376	1.9350	993.4797	1006.5631
beta116	1.4134	0.0525	1.0383	1.7779	996.8762	1003.1336
A1	1.7644	0.0234	1.4958	1.9977	996.9983	1003.0107
A2	1.7822	0.0729	1.0768	1.9979	998.8691	1001.1322
A3	1.7734	0.0821	0.8573	1.9997	999.8671	1000.1329
A4	1.8191	0.1144	0.6780	1.9999	999.2464	1000.7542
A5	1.8606	0.0201	1.6103	1.9989	996.4303	1003.5825
A6	1.9086	0.0059	1.7693	1.9965	992.0971	1007.9659
A7	1.7770	0.1615	0.8988	1.9976	999.2293	1000.7713
A8	1.8761	0.0148	1.7135	1.9970	996.0987	1003.9166
A9	1.6738	0.2072	0.5761	1.9930	1000.1105	999.8895
A10	1.8104	0.0491	1.3130	1.9993	997.6138	1002.3919
A11	1.6922	0.1651	0.7927	1.9938	999.3903	1000.6101
A12	1.8557	0.0604	1.6179	1.9997	998.1936	1001.8097
A13	1.7164	0.0957	1.2022	1.9933	998.2787	1001.7243
A14	1.6782	0.0936	1.2570	1.9940	997.9630	1002.0412
A15	1.8567	0.0212	1.6756	1.9975	992.2853	1007.7747
A16	1.8078	0.0807	1.4130	1.9947	999.9078	1000.0922
A17	1.8403	0.0092	1.7026	1.9987	991.0131	1009.0684
A18	1.8408	0.0263	1.4323	1.9921	995.6772	1004.3416
A19	1.8485	0.0266	1.5098	1.9975	995.2878	1004.7345
A20	1.7769	0.0694	1.1804	1.9997	998.3191	1001.6837
A21	1.8850	0.0221	1.7136	1.9990	991.4645	1008.6090
A22	1.8528	0.0126	1.6575	1.9995	980.5711	1019.8139

Continued on next page

Table E.6 – *Continued from previous page*

tree	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
A23	1.7740	0.1219	0.9551	1.9837	999.7947	1000.2053
A24	1.8230	0.0484	1.5118	1.9983	998.7862	1001.2153
A25	1.6834	0.1124	1.1506	2.0000	998.9965	1001.0045
A26	1.4838	0.4104	0.1644	1.9991	1000.6003	999.4001
A27	1.7397	0.0950	1.2567	1.9900	997.0398	1002.9690
A28	1.2107	0.5651	0.2012	1.9330	1000.7441	999.2565
A29	1.9284	0.0083	1.7457	2.0000	963.4873	1037.8964
A30	0.3417	0.0172	0.1448	0.6135	996.6759	1003.3352
A31	-0.0586	0.0318	-0.2984	0.3235	997.8419	1002.1628
A32	0.2430	0.0159	-0.1036	0.4177	994.2281	1005.8054
A33	0.3577	0.0960	-0.2203	0.8061	999.3158	1000.6847
A34	0.4940	0.0376	0.0184	0.9087	995.2459	1004.7768
A35	0.1226	0.0417	-0.1397	0.5977	997.4083	1002.5984
A36	-0.3289	0.0601	-0.6931	0.1002	997.5145	1002.4917
A37	-0.0759	0.1000	-0.6154	0.3508	999.2081	1000.7925
A38	-0.4095	0.0605	-0.7667	-0.0418	999.0675	1000.9334
A39	0.0675	0.0148	-0.1364	0.2702	993.5772	1006.4643
A40	-0.1520	0.0292	-0.4175	0.1255	996.7303	1003.2804
A41	-0.1416	0.0229	-0.3835	0.2029	996.1523	1003.8626
A42	-0.0013	0.0556	-0.4119	0.3272	998.4680	1001.5344
A43	-0.2075	0.0646	-0.5709	0.3224	999.8852	1000.1148
A44	-0.5779	0.0423	-0.9273	-0.1591	996.8377	1003.1723
A45	0.5136	0.0375	0.0369	0.7252	998.3509	1001.6518
A46	-0.0079	0.0373	-0.3035	0.2530	996.0039	1004.0121
A47	-0.1298	0.0237	-0.3758	0.1531	995.6793	1004.3394
A48	-0.2921	0.0487	-0.6053	0.1077	998.7314	1001.2702
A49	-0.0714	0.0490	-0.3794	0.3120	999.0830	1000.9178
A50	0.2585	0.0358	-0.1176	0.5444	997.5242	1002.4819
A51	-0.5434	0.0836	-0.9723	-0.0653	1000.0749	999.9251
A52	0.2133	0.0548	-0.0924	0.6309	997.5593	1002.4467
A53	0.1414	0.0282	-0.1996	0.3943	998.1339	1001.8696
A54	-0.1019	0.0190	-0.2835	0.1484	992.4864	1007.5705
A55	-0.0253	0.0189	-0.2524	0.2362	990.3774	1009.7161
A56	-0.1454	0.0098	-0.3050	0.0596	992.8077	1007.2444
A57	-0.3025	0.0335	-0.5290	0.0157	997.8793	1002.1252
A58	0.0762	0.0286	-0.2466	0.3924	997.3857	1002.6212
A59	-0.0990	0.0665	-0.4619	0.3598	998.8698	1001.1315
A60	0.3257	0.0512	-0.1218	0.6307	998.3397	1001.6631
A61	-0.0350	0.3019	-0.8026	0.6966	1000.3951	999.6051
A62	0.2536	0.0408	-0.1059	0.6317	998.5511	1001.4510
A63	0.1384	0.0268	-0.0893	0.3825	997.2476	1002.7600
A64	-0.2834	0.0523	-0.6486	0.0875	999.0773	1000.9236
A65	0.1660	0.0456	-0.1852	0.5219	998.7582	1001.2433
A66	-0.0506	0.0350	-0.3910	0.2667	999.2563	1000.7443
A67	-0.0830	0.0591	-0.5122	0.3802	999.3874	1000.6130
A68	-0.3440	0.0224	-0.5962	-0.1158	995.9976	1004.0185
A69	-0.1604	0.0186	-0.4101	0.0491	996.4179	1003.5950
A70	-0.1051	0.0584	-0.3882	0.2682	999.0529	1000.9480
A71	0.1529	0.0411	-0.2480	0.4738	998.8664	1001.1349
A72	-0.1606	0.0241	-0.4048	0.1276	996.8076	1003.2026
A73	-0.1452	0.0761	-0.5986	0.3018	1000.0993	999.9007
A74	-0.0013	0.0270	-0.2369	0.3429	998.7384	1001.2632
A75	0.0777	0.0189	-0.1996	0.2870	997.3394	1002.6677
A76	-0.0373	0.0218	-0.3285	0.1442	996.3580	1003.6553

Continued on next page

Table E.6 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
A77	0.1274	0.0228	-0.0834	0.4252	997.2962	1002.7111
A78	-0.6004	0.0317	-0.8260	-0.1062	998.0285	1001.9754
A79	0.2757	0.0229	0.0894	0.5240	992.6288	1007.4259
A80	-0.4354	0.0310	-0.7204	-0.0970	999.6564	1000.3437
A81	-0.0463	0.0464	-0.3984	0.2520	1000.0952	999.9048
A82	-0.0676	0.0461	-0.5704	0.3194	997.6168	1002.3889
A83	-0.3185	0.0688	-0.7874	-0.0649	999.9186	1000.0814
A84	-0.3938	0.0383	-0.6714	-0.0485	994.8170	1005.2100
A85	0.3138	0.0312	0.0054	0.6913	997.8529	1002.1517
A86	-0.1446	0.0509	-0.6080	0.1561	999.3502	1000.6502
A87	0.0647	0.0645	-0.4513	0.4627	998.6022	1001.3998
A88	0.2645	0.0315	-0.1162	0.4913	996.2056	1003.8089
A89	0.0736	0.0185	-0.0612	0.3716	996.4231	1003.5897
A90	-0.5157	0.0852	-0.8210	0.0106	999.9524	1000.0476
A91	0.0166	0.0263	-0.3175	0.2659	997.8104	1002.1944
A92	-0.4912	0.0691	-0.9501	-0.0891	998.7130	1001.2887
A93	0.3205	0.0346	0.0310	0.6537	995.6480	1004.3710
A94	0.0390	0.0182	-0.1775	0.2567	995.2980	1004.7242
A95	0.0770	0.0179	-0.1261	0.3602	997.1725	1002.8355
A96	0.1406	0.0593	-0.2303	0.4957	997.8449	1002.1598
A97	-0.1666	0.0620	-0.6354	0.2396	998.8939	1001.1073
A98	-0.1268	0.0206	-0.3897	0.1170	995.8732	1004.1439
A99	-0.0100	0.0505	-0.3221	0.3920	996.5940	1003.4176
A100	0.3193	0.0227	-0.0988	0.5353	995.2628	1004.7597
A101	0.3581	0.0260	0.0349	0.6769	994.4705	1005.5602
A102	-0.4156	0.0531	-0.6691	0.0891	997.5309	1002.4752
A103	-0.4079	0.0190	-0.7498	-0.1928	996.2912	1003.7226
A104	0.0498	0.0155	-0.1530	0.3464	992.9484	1007.1017
A105	0.1608	0.0289	-0.1517	0.4827	997.0764	1002.9322
A106	-0.1288	0.0911	-0.7116	0.3075	998.2982	1001.7047
A107	-0.1361	0.0278	-0.4681	0.0706	996.4893	1003.5231
A108	0.1340	0.0324	-0.1802	0.4129	996.3908	1003.6223
A109	0.1217	0.0452	-0.2404	0.4773	999.5913	1000.4089
A110	-0.0754	0.0219	-0.3396	0.1681	984.8835	1015.3485
A111	-0.1842	0.0268	-0.4997	0.1677	995.7575	1004.2606
A112	-0.0688	0.0303	-0.3396	0.3574	997.7101	1002.2952
A113	0.1868	0.0839	-0.3154	0.6729	1000.0981	999.9019
A114	-0.1922	0.0254	-0.3996	0.1428	997.9082	1002.0962
A115	-0.0383	0.0680	-0.7727	0.3366	998.1716	1001.8317
A116	-0.3640	0.0650	-0.7452	0.0923	998.5350	1001.4671
A117	0.1219	0.0585	-0.2708	0.5498	999.0057	1000.9953
A118	-0.0253	0.0634	-0.4650	0.3585	998.2082	1001.7950
A119	0.4224	0.0773	-0.1312	0.7630	997.7604	1002.2446
A120	0.2990	0.0244	-0.0122	0.5790	998.1761	1001.8272
A121	0.1636	0.0242	-0.1261	0.4391	996.2581	1003.7560
A122	-0.5403	0.0296	-0.8105	-0.2308	997.1813	1002.8267
A123	0.2781	0.0164	0.0966	0.5230	993.1273	1006.9203
A124	-0.1229	0.0216	-0.2974	0.1837	992.6210	1007.4339
A125	0.2181	0.0550	-0.1472	0.6132	999.2273	1000.7733
A126	0.1024	0.0471	-0.2913	0.4895	997.0984	1002.9100
A127	0.0548	0.0294	-0.2403	0.2890	992.7968	1007.2555
A128	-0.1661	0.0320	-0.4197	0.2325	996.6790	1003.3321
A129	-0.4001	0.0992	-0.8314	0.1220	999.7364	1000.2637
A130	-0.0134	0.0348	-0.2344	0.3410	997.7325	1002.2727

Continued on next page

Table E.6 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
A131	0.1910	0.0168	-0.0426	0.3817	995.5693	1004.4504
A132	-0.3323	0.0798	-0.9697	-0.0134	998.9246	1001.0766
A133	0.0886	0.0170	-0.1757	0.3054	993.0563	1006.9923
A134	-0.0885	0.0197	-0.3328	0.1156	997.5719	1002.4340
A135	-0.2568	0.1100	-0.6738	0.2582	999.5489	1000.4513
A136	0.1009	0.0681	-0.3533	0.4633	997.6984	1002.3069
A137	0.2561	0.0277	-0.0141	0.5578	997.7550	1002.2501
A138	0.0081	0.0378	-0.2583	0.3350	999.0875	1000.9133
A139	0.1470	0.0577	-0.2626	0.4786	998.8120	1001.1894
A140	-0.1090	0.0181	-0.3339	0.1164	993.6689	1006.3714
A141	-0.0892	0.0314	-0.3717	0.2221	996.2764	1003.7375
A142	0.1373	0.0302	-0.1057	0.5626	996.8617	1003.1482
A143	-0.0852	0.0182	-0.3906	0.1453	993.8543	1006.1837
A144	0.2569	0.0250	-0.0630	0.5439	995.3421	1004.6797
A145	-0.2957	0.0998	-0.6619	0.3076	1000.0068	999.9932
A146	-0.1965	0.0347	-0.4600	0.1084	998.0911	1001.9126
A147	0.2341	0.0626	-0.2078	0.5535	999.2072	1000.7934
A148	0.1591	0.0732	-0.3126	0.7584	998.2238	1001.7794
A149	-0.5441	0.0375	-0.8095	-0.1452	997.7629	1002.2421
A150	0.0624	0.0347	-0.2652	0.4491	998.2066	1001.7966
A151	-0.0005	0.0657	-0.4417	0.4546	998.5098	1001.4924
A152	-0.1412	0.0231	-0.4958	0.0877	995.4619	1004.5588
A153	-0.1879	0.0686	-0.6288	0.2200	999.3122	1000.6883
A154	-0.1729	0.0450	-0.5292	0.1506	998.4433	1001.5591
A155	0.0394	0.0240	-0.2305	0.3416	997.4982	1002.5081
A156	-0.0226	0.0430	-0.4572	0.2349	997.8994	1002.1050
A157	-0.1670	0.0453	-0.4828	0.2372	997.0150	1002.9939
A158	0.0688	0.0436	-0.1470	0.5005	998.6030	1001.3990
A159	0.2207	0.0201	-0.0401	0.4857	992.8353	1007.2164
A160	0.2448	0.0606	-0.1739	0.6854	998.4892	1001.5131
A161	-0.1977	0.0777	-0.6451	0.2373	999.6003	1000.3999
A162	0.0370	0.0207	-0.2559	0.2379	995.3114	1004.7107
A163	-0.4782	0.0739	-0.7974	0.0113	998.8124	1001.1890
A164	-0.0472	0.0278	-0.4560	0.1992	998.4015	1001.6011
A165	0.1675	0.0562	-0.2217	0.5089	999.3515	1000.6489
A166	0.1136	0.0143	-0.1066	0.3082	994.8317	1005.1951
A167	-0.0478	0.0566	-0.3851	0.3252	999.4436	1000.5567
A168	0.0117	0.0277	-0.2559	0.4081	997.6474	1002.3581
A169	-0.3719	0.0218	-0.5236	-0.0340	999.2092	1000.7914
A170	0.0272	0.0279	-0.2746	0.2281	997.6165	1002.3892
A171	0.0047	0.0565	-0.2953	0.4758	997.8327	1002.1720
A172	-0.2575	0.0334	-0.4901	0.1158	997.9383	1002.0660
A173	-0.2365	0.1054	-0.6853	0.3037	999.4516	1000.5487
A174	-0.1195	0.0500	-0.4331	0.3924	998.7557	1001.2459
A175	-0.0883	0.0187	-0.2925	0.1975	997.2749	1002.7325
A176	0.1134	0.0283	-0.1883	0.4580	997.8862	1002.1183
A177	0.3689	0.0773	-0.0525	0.7504	1000.3259	999.6742
A178	-0.2435	0.0165	-0.4431	0.0319	997.0736	1002.9350
A179	0.2108	0.1343	-0.2342	0.7876	1000.0336	999.9664
A180	0.1565	0.0725	-0.2231	0.6442	998.1576	1001.8458
A181	0.0811	0.0321	-0.0837	0.4327	997.9715	1002.0326
A182	0.0608	0.0444	-0.4128	0.4315	995.8352	1004.1822
A183	0.2366	0.0265	-0.0291	0.4753	993.0358	1007.0130
A184	-0.0562	0.0172	-0.2875	0.1663	995.5011	1004.5192

Continued on next page

Table E.6 – *Continued from previous page*

tree	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
A185	0.0858	0.0286	-0.2072	0.3379	998.1781	1001.8252
A186	0.0483	0.0829	-0.6923	0.4283	998.9724	1001.0287
A187	0.1373	0.0314	-0.1315	0.4904	996.5378	1003.4742
A188	0.2835	0.1369	-0.3126	0.7234	999.6728	1000.3273
A189	0.3262	0.0312	0.0099	0.5919	996.4882	1003.5242
A190	0.0196	0.0495	-0.4614	0.3526	997.8912	1002.1133
A191	-0.2104	0.0602	-0.5239	0.3500	997.8166	1002.1882
A192	-0.0619	0.0770	-0.4719	0.5132	998.9349	1001.0662
A193	-0.1168	0.0134	-0.2671	0.1463	993.6605	1006.3799
A194	-0.0163	0.0570	-0.3860	0.3112	998.7942	1001.2073
A195	0.4502	0.0407	-0.0474	0.6998	997.0579	1002.9508
A196	0.1868	0.0499	-0.1782	0.5148	997.9696	1002.0345
A197	-0.1011	0.0228	-0.3570	0.1820	996.1318	1003.8832
A198	-0.2987	0.0585	-0.6378	0.1645	998.5189	1001.4833
A199	-0.1896	0.0452	-0.5829	0.2291	997.2951	1002.7122
A200	-0.2704	0.0468	-0.6239	0.2201	999.3567	1000.6437
A201	0.1495	0.1160	-0.3256	0.7020	1000.2395	999.7606
A202	-0.1446	0.0629	-0.5457	0.3037	999.6156	1000.3845
A203	-0.3449	0.0377	-0.6211	0.0348	998.3841	1001.6185
A204	-0.2678	0.0403	-0.6400	0.0126	997.3423	1002.6648
A205	-0.0349	0.0343	-0.4351	0.2369	998.6146	1001.3873
A206	0.1663	0.0234	-0.1015	0.4648	994.6543	1005.3744
A207	-0.1805	0.0169	-0.3893	0.0516	995.8526	1004.1647
A208	0.4936	0.0710	-0.0099	0.8169	998.8906	1001.1106
A209	0.2813	0.0119	0.0746	0.4596	997.2240	1002.7837
A210	0.0528	0.0781	-0.4142	0.4087	999.8289	1000.1711
A211	0.3872	0.1145	-0.3478	0.8510	999.9858	1000.0142
A212	0.3447	0.0083	0.1693	0.5114	990.3966	1009.6965
A213	-0.0841	0.0236	-0.3190	0.1821	996.6947	1003.3163
A214	-0.2810	0.0387	-0.5293	0.1267	998.1879	1001.8154
A215	0.2960	0.0225	0.0227	0.5837	992.5689	1007.4867
A216	-0.0762	0.0221	-0.3599	0.1233	993.2546	1006.7912
A217	0.2687	0.1096	-0.2288	0.8618	999.8051	1000.1949
A218	-0.1446	0.1549	-0.9010	0.4834	1000.2589	999.7412
A219	0.1602	0.0132	-0.0419	0.4011	995.5111	1004.5091
A220	-0.3074	0.0592	-0.6395	0.1872	998.7264	1001.2752
A221	-0.4696	0.0710	-0.8434	0.0104	999.4876	1000.5127
A222	0.0186	0.1380	-0.6561	0.3769	1000.3630	999.6371
A223	-0.3803	0.0955	-0.7247	0.1093	999.5377	1000.4625
A224	-0.1734	0.0171	-0.3328	0.1044	997.5457	1002.4603
A225	-0.1701	0.0258	-0.4013	0.0494	995.1782	1004.8452
A226	-0.1859	0.0490	-0.4510	0.2340	998.6800	1001.3217
A227	0.3067	0.0619	-0.1570	0.5694	999.7956	1000.2044
A228	0.0986	0.0209	-0.1640	0.3339	995.5437	1004.4762
A229	0.3792	0.0412	0.0032	0.7173	997.4208	1002.5859
A230	0.2345	0.0392	-0.0299	0.6280	998.0711	1001.9326
A231	-0.1222	0.0060	-0.2482	0.0688	992.5450	1007.5110
A232	-0.0438	0.0220	-0.3656	0.1644	994.4112	1005.6202
A233	0.2509	0.0579	-0.1710	0.7205	999.1256	1000.8752
A234	0.0835	0.0312	-0.2046	0.3529	994.5184	1005.5118
A235	-0.4557	0.0608	-0.6856	0.0688	999.8153	1000.1847
A236	-0.0253	0.0272	-0.2552	0.2974	996.7363	1003.2744
A237	0.3202	0.0909	-0.2156	0.6959	999.9832	1000.0168
A238	0.1752	0.0169	-0.1453	0.3426	994.6623	1005.3663

Continued on next page

Table E.6 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
A239	0.1258	0.0510	-0.1044	0.6128	998.9365	1001.0646
A240	-0.2651	0.0300	-0.5087	0.1058	997.0185	1002.9904
A241	0.1177	0.0137	-0.0729	0.3674	992.3890	1007.6694
A242	-0.0085	0.0273	-0.2991	0.3041	996.2301	1003.7842
A243	-0.1267	0.0678	-0.6945	0.1455	999.3618	1000.6386
A244	0.0427	0.0664	-0.3026	0.5035	1000.1869	999.8131
A245	0.2439	0.0388	-0.0464	0.6549	997.6643	1002.3412
A246	-0.3550	0.1368	-0.8024	0.2092	999.8702	1000.1298
A247	-0.1208	0.0394	-0.4850	0.1564	998.1242	1001.8793
A248	-0.2021	0.0359	-0.5251	0.1155	996.4136	1003.5993
A249	-0.2588	0.0223	-0.4526	0.0805	997.7317	1002.2735
A250	0.1937	0.0789	-0.4517	0.4746	999.3494	1000.6510
A251	0.6533	0.1162	-0.0605	1.0876	1000.1715	999.8285
A252	-0.3334	0.0357	-0.6610	0.0080	998.6195	1001.3824
A253	-0.0177	0.0164	-0.2108	0.1876	989.1029	1011.0172
A254	0.1723	0.0178	-0.1524	0.3143	995.6381	1004.3810
A255	0.0012	0.0220	-0.1895	0.2707	996.3405	1003.6729
A256	0.0354	0.1061	-0.3630	0.7054	999.0041	1000.9969
A257	0.0827	0.0146	-0.1410	0.3012	993.2790	1006.7665
A258	0.0988	0.0410	-0.3268	0.3876	997.0863	1002.9222
A259	-0.0638	0.0394	-0.4996	0.2030	997.8242	1002.1805
A260	-0.1317	0.0546	-0.4094	0.3255	1000.1821	999.8179
A261	-0.1112	0.0212	-0.2993	0.1424	996.0869	1003.9285
A262	-0.3302	0.0540	-0.7260	0.0323	999.5660	1000.4342
A263	-0.0048	0.0196	-0.2230	0.2122	998.9895	1001.0115
A264	0.3168	0.0344	-0.0082	0.6385	999.2250	1000.7756
A265	-0.2463	0.0395	-0.5961	-0.0007	999.6039	1000.3963
A266	0.1431	0.0470	-0.1480	0.6038	999.4209	1000.5794
A267	-0.0627	0.0909	-0.4454	0.4759	999.8543	1000.1457
A268	-0.0351	0.0443	-0.4498	0.2642	999.1203	1000.8805
A269	-0.0881	0.0531	-0.4754	0.2710	999.8020	1000.1980
A270	0.0307	0.0134	-0.1609	0.2051	996.4854	1003.5270
A271	-0.1509	0.0693	-0.6534	0.2103	998.9131	1001.0881
A272	0.2716	0.0314	-0.0177	0.5601	998.3747	1001.6279
A273	-0.1032	0.0370	-0.5314	0.1730	998.6254	1001.3765
A274	0.0636	0.0566	-0.2021	0.5491	998.9954	1001.0056
A275	0.1304	0.0322	-0.2158	0.4892	998.0614	1001.9424
A276	-0.2882	0.0320	-0.5208	0.0689	998.3741	1001.6285
A277	-0.4117	0.0403	-0.7743	-0.0536	999.3892	1000.6112
A278	0.1444	0.0412	-0.1816	0.5522	998.6750	1001.3268
A279	-0.0867	0.0214	-0.3347	0.0832	999.3234	1000.6771
A280	0.2828	0.0385	-0.0677	0.5721	999.0434	1000.9575
A281	-0.1169	0.0602	-0.4651	0.3008	999.9302	1000.0698
A282	-0.2100	0.0459	-0.5172	0.1347	998.7879	1001.2136
A283	0.1517	0.0165	-0.1220	0.3237	995.3577	1004.6640
A284	0.3065	0.0632	-0.2316	0.7689	998.8154	1001.1860
A285	0.1358	0.0521	-0.2856	0.4023	999.2009	1000.7997
A286	-0.0857	0.1042	-0.5237	0.5231	998.8682	1001.1331
A287	0.3030	0.0877	-0.1793	0.7756	999.1373	1000.8634
A288	-0.1679	0.0418	-0.4535	0.1576	998.1446	1001.8588
A289	-0.0812	0.0440	-0.4288	0.2697	999.2428	1000.7578
A290	-0.0716	0.0232	-0.3662	0.2399	991.2455	1008.8318
A291	0.0604	0.2442	-0.5239	0.8967	1000.5626	999.4377
A292	-0.3966	0.0985	-0.7435	0.2117	999.2924	1000.7081

Continued on next page

Table E.6 – *Continued from previous page*

tree	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
A293	0.0737	0.0219	-0.2622	0.2500	995.3746	1004.6469
A294	-0.3023	0.0205	-0.5531	-0.0638	994.2966	1005.7361
A295	-0.1269	0.0451	-0.5350	0.0988	998.3854	1001.6172
A296	-0.0842	0.0400	-0.3494	0.3474	997.8217	1002.1831
A297	-0.0812	0.0458	-0.5529	0.1493	998.5619	1001.4402
A298	0.0114	0.0224	-0.5111	0.1982	992.2020	1007.8593
A299	-0.2249	0.0284	-0.4642	0.1433	997.0043	1003.0047
A300	0.0894	0.0416	-0.1688	0.4105	998.3284	1001.6744
A301	0.3168	0.0896	-0.1834	0.6970	999.4618	1000.5385
A302	-0.1675	0.0584	-0.5356	0.2792	999.0966	1000.9042
A303	-0.0232	0.0505	-0.4001	0.4042	996.2808	1003.7331
A304	0.3068	0.0424	-0.1125	0.6235	995.9672	1004.0491
A305	-0.0872	0.0285	-0.3652	0.2433	996.9587	1003.0506
A306	-0.1080	0.0315	-0.4436	0.1749	998.2581	1001.7449
A307	0.0064	0.0190	-0.2328	0.2260	989.7982	1010.3069
A308	-0.2706	0.0814	-0.6776	0.2542	1000.0974	999.9026
A309	0.0283	0.1413	-0.3648	0.7302	1000.2941	999.7060
A310	-0.1291	0.0450	-0.4455	0.1773	999.5964	1000.4038
A311	0.1566	0.0210	-0.1524	0.3767	990.7581	1009.3281
A312	0.2135	0.0921	-0.1736	0.7314	999.9088	1000.0912
A313	-0.2065	0.0726	-0.5427	0.2540	999.5657	1000.4345
A314	0.0638	0.0066	-0.0823	0.2131	990.0230	1010.0775
A315	-0.0524	0.0615	-0.3914	0.3229	999.2923	1000.7082
A316	0.1241	0.0338	-0.1475	0.4594	998.3121	1001.6908
A317	-0.2135	0.0418	-0.4625	0.1643	998.9794	1001.0216
A318	0.4253	0.0465	0.1331	0.8875	995.5802	1004.4394
A319	-0.3658	0.0432	-0.7289	-0.0557	998.2410	1001.7621
A320	-0.1629	0.0308	-0.4630	0.0830	996.5058	1003.5065
A321	0.4462	0.0411	0.0134	0.6607	999.0520	1000.9489
A322	0.2092	0.0356	-0.2596	0.4617	996.3899	1003.6232
A323	-0.4434	0.0264	-0.6063	-0.0759	996.0280	1003.9878
A324	-0.2964	0.0243	-0.5306	0.0032	996.4350	1003.5778
A325	-0.2379	0.0393	-0.5645	0.0847	998.4577	1001.5447
A326	-0.0702	0.0306	-0.3138	0.2759	995.5529	1004.4670
A327	-0.1271	0.0704	-0.6377	0.2007	999.9573	1000.0427
A328	-0.0208	0.0321	-0.3511	0.2493	996.3651	1003.6482
A329	-0.0559	0.0327	-0.4597	0.1342	999.3033	1000.6972
A330	-0.0077	0.1361	-0.4550	0.5691	1000.3144	999.6857
A331	0.2626	0.0233	-0.1122	0.5349	996.1461	1003.8688
A332	-0.0680	0.0380	-0.3014	0.4642	996.7342	1003.2765
A333	-0.3126	0.1334	-0.8346	0.3739	999.1962	1000.8044
A334	0.3373	0.0121	0.1232	0.5080	990.1021	1009.9968
A335	-0.1815	0.0361	-0.4268	0.1823	998.3327	1001.6701
A336	0.1888	0.0405	-0.1411	0.4886	995.6316	1004.3876
A337	-0.6782	0.0747	-0.9845	-0.0325	1000.5545	999.4458
A338	-0.2705	0.0348	-0.4947	0.1134	998.0883	1001.9154
A339	-0.2746	0.1056	-0.6066	0.5332	999.9951	1000.0049
A340	-0.2784	0.0119	-0.4934	-0.0957	996.8217	1003.1884
A341	-0.0581	0.0399	-0.4113	0.1794	999.4985	1000.5018
A342	-0.4178	0.0383	-0.6489	-0.0300	994.7184	1005.3096
A343	-0.0660	0.0232	-0.3723	0.2421	994.5515	1005.4783
A344	0.1981	0.0434	-0.1601	0.5312	998.0717	1001.9320
A345	-0.2721	0.1328	-0.8141	0.3870	1000.0872	999.9128
A346	-0.0374	0.0888	-0.4072	0.7434	998.9690	1001.0321

Continued on next page

Table E.6 – *Continued from previous page*

tree	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
A347	0.2803	0.0175	0.0839	0.5451	994.5779	1005.4517
A348	0.0436	0.0590	-0.2367	0.5803	998.9678	1001.0333
A349	0.0917	0.0548	-0.2887	0.4596	998.5287	1001.4735
A350	0.0115	0.0330	-0.2846	0.2428	998.8627	1001.1386
A351	0.3443	0.0208	0.0622	0.5754	998.1962	1001.8071
A352	-0.0858	0.0105	-0.2714	0.1026	992.7784	1007.2741
A353	-0.1424	0.0182	-0.3679	0.0679	994.2310	1005.8025
A354	0.1275	0.0554	-0.2628	0.5389	998.6332	1001.3687
A355	-0.1851	0.0303	-0.4140	0.1187	998.5693	1001.4327
A356	0.0138	0.0226	-0.2540	0.2904	996.7480	1003.2626
A357	0.1926	0.1126	-0.4076	0.8487	999.7274	1000.2727
A358	0.1810	0.0373	-0.1287	0.4976	997.9019	1002.1025
A359	-0.2418	0.0853	-0.6811	0.2744	999.5079	1000.4923
A360	-0.1970	0.0384	-0.4833	0.1374	998.2803	1001.7227
A361	0.3212	0.0321	-0.0495	0.6252	997.3965	1002.6103
A362	-0.1409	0.0388	-0.4841	0.1875	998.7471	1001.2545
A363	-0.1618	0.0214	-0.3950	0.0719	996.0944	1003.9209
A364	-0.1796	0.0279	-0.3817	0.1332	998.1302	1001.8733
A365	0.4522	0.0446	0.1416	0.8639	997.8862	1002.1183
A366	-0.1953	0.0365	-0.5222	0.0616	995.3250	1004.6970
A367	-0.1519	0.0314	-0.3976	0.1778	995.8068	1004.2109
A368	-0.0747	0.0109	-0.3597	0.0427	993.3757	1006.6685
A369	-0.1856	0.1128	-0.5734	0.3093	999.9676	1000.0324
A370	-0.2792	0.0635	-0.6470	0.3217	999.5728	1000.4274
A371	0.1248	0.0451	-0.2184	0.5920	997.4138	1002.5929
A372	-0.0555	0.0554	-0.4871	0.3319	998.0099	1001.9941
A373	0.2283	0.0070	0.0366	0.3517	987.7024	1012.4507
A374	0.3186	0.0396	-0.0515	0.7611	997.5146	1002.4916
A375	0.1321	0.0817	-0.3226	0.7038	999.1226	1000.8782
A376	0.2851	0.2144	-0.5023	0.8548	1000.3906	999.6096
A377	-0.0382	0.0369	-0.4241	0.2583	995.1218	1004.9021
A378	-0.1620	0.0526	-0.5372	0.1918	997.9380	1002.0663
A379	0.0890	0.0087	-0.1135	0.2272	994.0805	1005.9547
A380	0.0214	0.0380	-0.2954	0.3481	997.5355	1002.4706
A381	-0.0775	0.0189	-0.3416	0.1142	992.4170	1007.6409
A382	0.0890	0.0781	-0.3520	0.5748	998.4200	1001.5825
A383	0.0422	0.0066	-0.1476	0.1336	993.1877	1006.8590
A384	-0.0170	0.0294	-0.3461	0.2987	995.0732	1004.9512
A385	0.1127	0.0189	-0.1411	0.4886	994.4727	1005.5580
A386	-0.1345	0.0318	-0.9845	-0.0325	995.7219	1004.2965
A387	-0.2658	0.0536	-0.4947	0.1134	999.4113	1000.5890
A388	0.4389	0.0569	-0.6066	0.5332	999.3225	1000.6780
A389	-0.0574	0.0225	-0.4934	-0.0957	998.6688	1001.3330
A390	-0.7343	0.0388	-0.4113	0.1794	998.7653	1001.2362
A391	0.0902	0.0377	-0.6489	-0.0300	997.2771	1002.7303
A392	-0.1927	0.0497	-0.3723	0.2421	997.8284	1002.1763
A393	0.0551	0.0226	-0.1601	0.5312	997.2691	1002.7384
A394	-0.1223	0.0612	-0.8141	0.3870	998.1471	1001.8563
A395	-0.1075	0.0490	-0.4072	0.7434	998.4951	1001.5072
A396	-0.0226	0.0190	0.0839	0.5451	997.7848	1002.2201
A397	0.1398	0.0506	-0.2367	0.5803	998.4509	1001.5515
A398	-0.0688	0.0329	-0.2887	0.4596	999.3426	1000.6578
A399	0.0487	0.0270	-0.2846	0.2428	996.8639	1003.1460
A400	0.2780	0.0460	0.0622	0.5754	996.8916	1003.1181

Continued on next page

Table E.6 – *Continued from previous page*

tree	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
A401	0.2147	0.0328	-0.2714	0.1026	997.2936	1002.7137
A402	-0.1581	0.0379	-0.3679	0.0679	994.8681	1005.1584
A403	0.0846	0.0430	-0.2628	0.5389	997.6863	1002.3191
A404	-0.1526	0.0167	-0.4140	0.1187	997.5586	1002.4474
A405	-0.3469	0.0590	-0.2540	0.2904	998.6782	1001.3235
A406	-0.2477	0.0328	-0.4076	0.8487	998.2841	1001.7188
A407	-0.1834	0.0364	-0.1287	0.4976	998.2536	1001.7495
A408	0.2702	0.0087	-0.6811	0.2744	994.8555	1005.1711
A409	0.4279	0.1386	-0.4833	0.1374	999.9641	1000.0359
A410	0.2623	0.0791	-0.0495	0.6252	999.4869	1000.5134
A411	-0.1001	0.0516	-0.4841	0.1875	999.1925	1000.8082
A412	0.1188	0.0172	-0.3950	0.0719	998.5523	1001.4498
A413	-0.3835	0.0321	-0.3817	0.1332	999.6034	1000.3968
A414	-0.0046	0.0200	0.1416	0.8639	996.1183	1003.8968
A415	0.1328	0.0274	-0.5222	0.0616	998.4345	1001.5680
A416	0.1537	0.0506	-0.3976	0.1778	999.5540	1000.4462
A417	0.1218	0.0727	-0.3597	0.0427	999.9300	1000.0700
A418	0.0789	0.0202	-0.5734	0.3093	995.5739	1004.4458
A419	0.0198	0.0098	-0.6470	0.3217	994.2113	1005.8224
A420	0.3374	0.0383	-0.2184	0.5920	999.7188	1000.2813
A421	-0.0096	0.0248	-0.4871	0.3319	994.9306	1005.0952
A422	-0.1138	0.0310	0.0366	0.3517	997.0039	1003.0051
A423	0.3566	0.0412	-0.0515	0.7611	997.9240	1002.0803
A424	0.0976	0.0169	-0.3226	0.7038	996.6418	1003.3695
A425	-0.2248	0.0315	-0.5023	0.8548	994.8080	1005.2191
A426	0.0407	0.0134	-0.4241	0.2583	996.1638	1003.8510
A427	0.1155	0.0542	-0.5372	0.1918	999.7406	1000.2595
A428	0.1201	0.0634	-0.1135	0.2272	999.5336	1000.4666
A429	0.0530	0.0305	-0.2954	0.3481	997.1291	1002.8792
A430	-0.0040	0.0164	-0.3416	0.1142	994.8717	1005.1547
A431	-0.1502	0.0183	-0.3520	0.5748	997.4666	1002.5398
A432	0.0562	0.0552	-0.1476	0.1336	999.1890	1000.8117
A433	0.1390	0.0337	-0.3461	0.2987	999.1123	1000.8885
A434	-0.2295	0.0099	-0.4241	0.2583	999.1123	1000.8885
A435	0.1712	0.0073	-0.3817	0.6252	997.9240	1002.0803

Chapter 5

Table E.7: Combined GPU-hmmer parameter summary after 5,000,000 MCMC iterations for hidden states $K = 25$

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
posterior	-1.0987E+03	7.6624E+04	-1.6110E+03	-4.0624E+02	35830.8997	239.54
Gam1	8.6710E-01	2.4186E-04	8.3770E-01	8.9248E-01	29571.9564	169.07
Gam2	1.6289E-02	4.7111E-05	3.8358E-03	2.8083E-02	14455.9226	345.87
Gam3	1.8235E-03	8.1549E-06	5.3391E-06	5.9633E-03	31080.9139	160.87
Gam4	1.8604E-03	9.1692E-06	8.0933E-06	6.5181E-03	23964.1982	208.64
Gam5	1.4205E-03	1.2756E-06	4.4740E-05	3.3164E-03	14213.5729	351.77
Gam6	3.4741E-03	7.6753E-06	2.8037E-05	8.2833E-03	21277.4832	234.99
Gam7	2.9676E-03	3.5263E-06	4.7983E-05	5.7603E-03	15272.0127	327.39

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam8	1.9523E-03	5.1619E-06	1.8840E-05	4.8152E-03	27378.2848	182.62
Gam9	8.8053E-04	1.4839E-06	6.1831E-07	2.6187E-03	15509.9601	322.37
Gam10	4.0934E-03	1.6603E-05	8.1382E-15	1.2107E-02	57380.8895	87.137
Gam11	9.3274E-03	1.4699E-05	1.9398E-03	1.5410E-02	45068.2804	110.94
Gam12	6.1730E-04	3.1284E-07	4.1541E-07	1.7926E-03	14440.9950	346.23
Gam13	3.5715E-03	5.4052E-05	1.2686E-04	8.2704E-03	24168.9799	206.87
Gam14	9.6052E-04	7.1797E-07	8.2637E-06	2.5498E-03	18857.9341	265.14
Gam15	1.3255E-03	1.4142E-06	4.1034E-06	4.0367E-03	31384.2830	159.31
Gam16	1.5935E-03	1.1240E-05	7.3288E-06	4.9660E-03	20798.4511	240.40
Gam17	3.8764E-02	3.7291E-05	2.6857E-02	4.8477E-02	48946.9321	102.15
Gam18	1.0956E-03	1.9142E-06	7.6476E-06	3.7071E-03	45031.8581	111.03
Gam19	1.0625E-03	4.7366E-06	5.9749E-06	2.6002E-03	17168.5493	291.23
Gam20	2.6584E-03	5.6930E-06	2.0232E-05	6.8404E-03	35644.4221	140.27
Gam21	2.9296E-02	2.9234E-05	1.9465E-02	4.0877E-02	42258.6322	118.31
Gam22	1.9319E-03	3.1337E-06	8.6148E-05	6.0423E-03	16100.7869	310.54
Gam23	2.3521E-03	3.8152E-06	3.0464E-05	6.1173E-03	22249.4145	224.72
Gam24	2.8243E-03	7.0229E-06	1.4940E-04	7.1177E-03	15713.5026	318.19
Gam25	7.5430E-04	1.2261E-06	0.0000E+00	2.8100E-03	33773.0435	148.04
Gam26	7.5800E-04	9.2460E-07	2.7447E-06	2.1539E-03	19423.7468	257.41
Gam27	9.7455E-01	1.8772E-05	9.6572E-01	9.8144E-01	16385.9590	305.13
Gam28	1.1656E-03	5.5348E-07	3.8569E-06	2.4886E-03	433749.2599	11.527
Gam29	2.6091E-04	1.4724E-07	6.0330E-07	5.3802E-04	34952.5560	143.05
Gam30	1.1413E-03	2.5184E-07	3.8769E-04	1.6722E-03	12690.4915	393.99
Gam31	2.5300E-03	2.2597E-07	1.7208E-03	3.7603E-03	62747.9452	79.683
Gam32	7.2782E-04	6.3201E-07	8.2677E-05	1.5357E-03	22661.4152	220.63
Gam33	4.7610E-03	2.8677E-06	2.1785E-03	7.5344E-03	128377.9942	38.947
Gam34	1.4483E-04	3.3611E-07	0.0000E+00	4.4469E-04	18494.6559	270.34
Gam35	8.8254E-04	9.1352E-06	1.1430E-179	2.6530E-03	17104.6382	292.31
Gam36	2.2049E-04	5.6664E-07	3.2433E-07	8.8779E-04	13504.7449	370.24
Gam37	2.8423E-04	1.8785E-06	1.3566E-07	5.3904E-04	16705.1430	299.30
Gam38	1.8184E-03	8.9467E-07	1.0682E-03	2.9130E-03	12987.5478	384.98
Gam39	1.5827E-04	3.8431E-07	8.2793E-240	5.5617E-04	10408.8774	480.35
Gam40	1.5324E-04	5.9919E-08	1.7997E-06	3.9441E-04	35391.8500	141.27
Gam41	1.1195E-04	6.7680E-07	4.5519E-07	1.0529E-04	6175.3438	809.67
Gam42	1.6970E-04	5.8092E-08	2.3201E-68	4.8800E-04	19975.0180	250.31
Gam43	5.4145E-04	1.7014E-06	8.0357E-07	1.2248E-03	12214.7293	409.34
Gam44	9.4007E-05	1.2845E-07	0.0000E+00	3.0288E-04	15527.8979	322.00
Gam45	5.8817E-04	2.5773E-06	2.6848E-07	1.7851E-03	16773.6991	298.08
Gam46	2.7130E-04	1.6124E-06	0.0000E+00	9.7746E-04	12013.6336	416.19
Gam47	8.3608E-04	8.7118E-07	8.1343E-06	1.6621E-03	25000.7219	199.99
Gam48	6.1769E-03	1.6042E-06	3.7912E-03	8.2029E-03	72454.3676	69.009
Gam49	1.3814E-03	1.2691E-07	8.2454E-04	1.8834E-03	16414.5669	304.60
Gam50	2.7088E-04	1.8704E-07	1.9386E-06	8.3363E-04	20676.1070	241.82
Gam51	3.8265E-03	1.4101E-05	1.5164E-06	1.1256E-02	54832.1289	91.187
Gam52	4.1702E-02	2.7147E-04	1.4143E-02	7.0675E-02	177277.4811	28.204
Gam53	6.5895E-01	1.1852E-02	5.3762E-01	8.6165E-01	179266.1419	27.891
Gam54	5.2686E-02	1.3514E-04	3.0989E-02	7.5725E-02	20727.5986	241.22
Gam55	3.7342E-03	8.7625E-06	4.2153E-06	8.8395E-03	35075.5349	142.54
Gam56	3.7250E-03	1.4367E-05	2.1957E-05	1.0399E-02	13703.4629	364.87
Gam57	3.4035E-03	7.6310E-06	1.1752E-06	9.5530E-03	60398.7204	82.783
Gam58	1.1569E-02	5.7239E-05	1.4684E-03	2.6915E-02	126050.2525	39.666
Gam59	8.3384E-03	5.0755E-05	2.4430E-05	1.9246E-02	108972.2649	45.883
Gam60	7.5111E-03	1.8539E-05	1.3374E-03	1.5506E-02	35526.5778	140.73
Gam61	2.4433E-03	5.7828E-06	3.7794E-05	7.1885E-03	23869.7025	209.47

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam62	2.9865E-03	1.1750E-05	0.0000E+00	9.4605E-03	24301.5009	205.74
Gam63	4.6511E-03	1.1309E-05	4.0228E-04	1.1406E-02	17927.0292	278.90
Gam64	3.4551E-03	1.0811E-05	8.0421E-06	9.2846E-03	60480.2163	82.671
Gam65	1.7704E-03	2.9336E-06	2.0011E-05	5.5824E-03	31567.4252	158.39
Gam66	1.8257E-03	5.5040E-06	4.8305E-06	6.7129E-03	21359.3236	234.08
Gam67	4.4132E-03	1.0302E-05	2.3097E-04	1.0175E-02	30529.5466	163.77
Gam68	3.3537E-03	1.3710E-05	4.6203E-06	1.1097E-02	47189.7634	105.95
Gam69	2.5590E-03	1.2021E-05	1.6972E-05	8.6447E-03	17168.6412	291.22
Gam70	3.6075E-03	2.1892E-05	1.3317E-05	1.4296E-02	51779.6476	96.563
Gam71	2.7341E-03	1.0429E-05	6.6654E-05	9.0064E-03	50454.1729	99.099
Gam72	1.6084E-01	8.9549E-03	1.7265E-04	2.6376E-01	52563.5955	95.122
Gam73	4.7693E-03	1.3051E-05	9.0943E-04	1.3995E-02	31721.7813	157.62
Gam74	3.2346E-03	1.1070E-05	2.3274E-05	8.6432E-03	32056.6747	155.97
Gam75	1.9117E-03	3.3446E-06	0.0000E+00	5.6489E-03	15101.7754	331.08
Gam76	4.0374E-03	1.3862E-05	5.6963E-06	1.1135E-02	17448.4236	286.55
Gam77	1.8735E-02	1.3793E-04	1.4735E-03	3.7747E-02	153773.7616	32.515
Gam78	5.2840E-02	1.4351E-03	8.7997E-03	1.2569E-01	109036.5946	45.856
Gam79	7.7194E-01	4.7660E-04	7.3193E-01	8.1135E-01	97858.7582	51.094
Gam80	2.3710E-03	6.3079E-06	8.1711E-209	6.0531E-03	40191.5185	124.40
Gam81	9.8015E-03	2.1754E-05	2.7665E-03	1.9849E-02	14021.4458	356.59
Gam82	8.0723E-03	1.8660E-05	1.5913E-03	1.6722E-02	13912.5818	359.38
Gam83	4.2189E-03	1.5787E-05	5.4825E-05	1.2599E-02	17447.4497	286.57
Gam84	1.5088E-02	5.6666E-05	2.5035E-03	3.0542E-02	25580.4554	195.46
Gam85	9.6710E-03	1.6331E-05	2.9544E-03	1.8682E-02	31452.3565	158.97
Gam86	1.6917E-03	2.8601E-06	8.1084E-06	5.7956E-03	8240.6163	606.75
Gam87	5.7498E-03	2.7984E-05	2.2783E-04	1.5747E-02	21734.0305	230.05
Gam88	6.3461E-03	1.9530E-05	4.5535E-04	1.4662E-02	26108.6270	191.50
Gam89	5.2471E-03	1.3105E-05	2.0134E-04	1.0240E-02	13017.1322	384.10
Gam90	2.0111E-03	4.2089E-06	1.1327E-05	5.9145E-03	25705.1264	194.51
Gam91	1.0309E-03	1.6301E-06	0.0000E+00	3.8256E-03	28956.6968	172.67
Gam92	2.2777E-03	3.1269E-06	1.0209E-05	5.9816E-03	14923.2556	335.04
Gam93	2.1677E-03	9.3605E-06	0.0000E+00	5.6929E-03	23761.2254	210.42
Gam94	1.0397E-03	3.5146E-06	4.8823E-06	4.4321E-03	23742.1096	210.59
Gam95	3.3163E-03	5.2410E-06	1.4686E-04	7.9886E-03	30114.5464	166.03
Gam96	2.5090E-03	1.6163E-05	2.7992E-05	7.1467E-03	41015.2414	121.90
Gam97	5.8867E-02	1.3932E-03	2.0425E-05	1.0662E-01	31943.0005	156.52
Gam98	5.7913E-03	2.4696E-05	4.1050E-50	1.4469E-02	35391.9791	141.27
Gam99	1.4880E-03	2.3845E-06	0.0000E+00	4.5140E-03	25023.0598	199.81
Gam100	3.6941E-03	6.4817E-06	3.0497E-04	7.7939E-03	20250.6950	246.90
Gam101	1.0241E-02	7.2072E-05	1.0568E-04	2.7836E-02	20759.9054	240.84
Gam102	1.0804E-01	4.6584E-04	6.1602E-02	1.4489E-01	14058.9871	355.64
Gam103	5.6760E-03	2.0145E-05	5.4058E-06	1.4865E-02	82129.7028	60.879
Gam104	2.8431E-03	6.8754E-06	1.2424E-04	8.9155E-03	34299.2175	145.77
Gam105	7.1477E-01	7.0890E-04	6.6167E-01	7.6297E-01	11826.7772	422.76
Gam106	6.5209E-03	1.7024E-05	1.5508E-04	1.4459E-02	18234.4629	274.20
Gam107	2.8461E-02	2.2138E-04	7.8455E-03	5.4417E-02	17921.7664	278.99
Gam108	3.3291E-03	8.5009E-06	1.2591E-05	9.5390E-03	26847.8163	186.23
Gam109	1.0648E-02	4.2232E-05	3.1849E-04	2.3514E-02	37656.6887	132.77
Gam110	1.4558E-02	9.6667E-05	1.4384E-03	3.0609E-02	19327.2137	258.70
Gam111	2.6155E-03	1.2609E-05	0.0000E+00	1.0664E-02	35246.2265	141.85
Gam112	3.1530E-02	9.3212E-05	1.5939E-02	5.2513E-02	21814.3469	229.20
Gam113	3.0332E-03	9.1017E-06	2.1723E-05	8.8238E-03	21228.0396	235.53
Gam114	2.8459E-03	5.0074E-06	8.5689E-05	8.0017E-03	20882.2793	239.43
Gam115	3.2308E-03	1.2224E-05	5.1372E-05	9.8159E-03	26005.5177	192.26

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam116	1.7967E-03	3.4193E-06	5.5578E-06	5.4687E-03	14293.7529	349.80
Gam117	5.1239E-03	2.2861E-05	7.6876E-06	1.4478E-02	14718.0883	339.71
Gam118	3.9297E-03	1.7284E-05	0.0000E+00	1.0973E-02	30107.3335	166.07
Gam119	2.4660E-03	6.7519E-06	3.6767E-05	8.8143E-03	39887.9865	125.35
Gam120	5.9192E-03	2.0708E-05	2.1575E-05	1.4863E-02	41440.4025	120.65
Gam121	3.3092E-03	7.1388E-06	1.2865E-04	9.1760E-03	19966.6857	250.41
Gam122	1.4188E-02	1.4186E-04	1.0583E-04	3.8582E-02	29696.9340	168.36
Gam123	6.1469E-03	3.7944E-05	3.5337E-103	2.0600E-02	14266.5070	350.47
Gam124	5.3542E-03	6.3464E-05	2.1460E-05	1.3922E-02	16095.5744	310.64
Gam125	3.4203E-03	2.1335E-05	0.0000E+00	1.0130E-02	45807.5273	109.15
Gam126	1.4618E-02	1.1325E-04	2.8701E-04	3.6125E-02	13691.7749	365.18
Gam127	3.2206E-01	1.0197E-03	2.6087E-01	3.7722E-01	49086.4278	101.86
Gam128	7.2857E-03	3.1361E-05	1.3948E-04	1.8385E-02	42005.7489	119.03
Gam129	1.6891E-02	9.2670E-05	6.3445E-04	3.3494E-02	7378.6105	677.63
Gam130	4.6442E-03	1.3438E-05	1.1728E-04	1.1089E-02	13249.6606	377.36
Gam131	4.9890E-01	1.3868E-03	4.2268E-01	5.6160E-01	22054.7456	226.70
Gam132	8.1591E-03	2.6096E-05	2.4168E-04	1.6574E-02	11093.0986	450.73
Gam133	5.9883E-03	2.4691E-05	0.0000E+00	1.6412E-02	26398.9966	189.40
Gam134	7.2804E-03	2.6287E-05	8.8405E-05	1.8001E-02	18358.7256	272.35
Gam135	8.7256E-03	2.1526E-05	2.2706E-04	1.7155E-02	23918.9752	209.03
Gam136	2.9880E-03	7.3198E-06	2.0184E-06	7.9991E-03	19008.6479	263.03
Gam137	4.1896E-03	1.5155E-05	0.0000E+00	1.0915E-02	25792.6040	193.85
Gam138	4.0759E-03	1.7561E-05	0.0000E+00	1.2611E-02	26961.9908	185.44
Gam139	1.9013E-03	2.7532E-06	3.6100E-06	5.6579E-03	24439.0003	204.59
Gam140	3.0420E-03	9.0646E-06	0.0000E+00	9.7842E-03	13533.4384	369.45
Gam141	5.9292E-03	9.9349E-05	0.0000E+00	1.8168E-02	26232.0800	190.60
Gam142	3.5377E-03	9.6379E-06	0.0000E+00	1.0215E-02	23203.7127	215.48
Gam143	1.5168E-02	6.2250E-05	1.9025E-03	3.0656E-02	15722.1103	318.02
Gam144	5.7933E-03	4.4980E-05	1.7462E-289	1.6168E-02	22946.4046	217.89
Gam145	5.7463E-03	3.2700E-05	2.3855E-05	1.7080E-02	21825.2432	229.09
Gam146	2.6456E-03	1.2255E-05	2.5832E-06	9.3531E-03	24569.0253	203.50
Gam147	1.4579E-02	1.2048E-04	0.0000E+00	3.9936E-02	16737.9921	298.72
Gam148	8.6204E-03	5.7072E-05	2.7282E-06	2.0417E-02	38093.7914	131.25
Gam149	1.6937E-02	6.7864E-05	4.1349E-03	3.4704E-02	17925.6478	278.92
Gam150	1.0294E-02	4.6247E-05	1.4572E-03	2.1769E-02	18136.6081	275.68
Gam151	1.3152E-02	7.3651E-05	4.6341E-05	3.0154E-02	15556.8030	321.40
Gam152	3.7244E-02	2.8304E-04	2.8950E-03	6.5610E-02	33478.3899	149.35
Gam153	1.3758E-02	6.6246E-05	4.0890E-04	3.0621E-02	29346.9173	170.37
Gam154	5.9445E-03	2.1308E-05	1.4043E-04	1.5699E-02	32066.2635	155.92
Gam155	2.8742E-02	1.1982E-04	7.3737E-03	5.2091E-02	33898.0279	147.50
Gam156	2.5899E-03	7.0253E-06	7.4161E-06	7.8707E-03	18689.6816	267.52
Gam157	7.4292E-01	1.0465E-03	6.7896E-01	7.9747E-01	17523.3619	285.33
Gam158	3.7707E-03	1.0295E-05	0.0000E+00	8.9664E-03	39356.2073	127.04
Gam159	4.6426E-02	1.6495E-04	1.9526E-02	6.8351E-02	20248.3966	246.93
Gam160	9.4584E-03	2.2591E-05	2.0420E-03	1.9575E-02	15479.8442	323.00
Gam161	2.4250E-03	6.0102E-06	2.2032E-05	6.5940E-03	24814.9610	201.49
Gam162	1.1943E-02	5.1395E-05	1.7040E-03	2.6546E-02	19386.5635	257.91
Gam163	2.5954E-03	5.8674E-06	1.6756E-05	7.9368E-03	48028.9088	104.10
Gam164	2.1189E-03	4.6098E-06	6.1450E-06	5.7563E-03	21749.6357	229.88
Gam165	2.2330E-03	1.8865E-05	8.4136E-06	6.9105E-03	32478.0637	153.95
Gam166	3.6582E-03	8.4473E-06	9.5463E-05	9.4273E-03	22048.8286	226.76
Gam167	3.6427E-03	1.1416E-05	1.0440E-05	9.4321E-03	28140.4471	177.68
Gam168	1.1421E-02	6.9643E-05	4.6150E-04	2.6784E-02	25694.6466	194.59
Gam169	5.1223E-03	4.4866E-05	1.0712E-04	1.6117E-02	24383.3685	205.05

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam170	4.9605E-03	2.9550E-05	1.9398E-05	1.4858E-02	19684.0267	254.01
Gam171	2.9722E-03	1.0856E-05	0.0000E+00	7.2370E-03	19550.7361	255.74
Gam172	1.9379E-02	2.5865E-04	2.0458E-04	5.2602E-02	42800.9521	116.81
Gam173	1.3881E-02	7.6848E-05	6.7001E-04	3.1088E-02	23168.4207	215.81
Gam174	6.8947E-03	1.6571E-05	1.2776E-03	1.5726E-02	18513.8606	270.06
Gam175	2.7526E-03	4.9136E-06	1.9243E-63	7.1387E-03	36917.1373	135.43
Gam176	2.6324E-02	2.7210E-04	8.3009E-04	5.7787E-02	34809.0990	143.64
Gam177	5.5414E-01	2.2137E-03	4.7383E-01	6.4615E-01	47621.5790	104.99
Gam178	2.5632E-02	2.7222E-04	3.2199E-03	6.4055E-02	63701.0484	78.491
Gam179	1.3130E-02	9.7449E-05	8.0479E-04	3.2826E-02	19797.6125	252.55
Gam180	5.8898E-03	2.2534E-05	1.5316E-04	1.6794E-02	45691.9265	109.42
Gam181	7.4855E-03	2.9769E-05	7.2334E-04	1.9642E-02	46600.4791	107.29
Gam182	4.8547E-03	1.6843E-05	0.0000E+00	1.3644E-02	15670.4387	319.07
Gam183	8.8900E-02	1.2533E-03	2.7782E-02	1.5979E-01	55484.9668	90.114
Gam184	6.7210E-03	6.9651E-05	3.3967E-06	2.5447E-02	22950.5821	217.85
Gam185	6.8905E-03	2.4856E-05	2.6100E-05	1.6220E-02	23021.4924	217.18
Gam186	1.0912E-02	1.1490E-04	1.9169E-05	2.9730E-02	24510.3336	203.99
Gam187	7.3192E-03	7.6226E-05	8.4817E-07	2.4485E-02	59524.0225	83.999
Gam188	1.2526E-02	5.8547E-05	7.1780E-04	2.5975E-02	18057.2891	276.89
Gam189	7.0914E-03	4.2373E-05	0.0000E+00	1.9989E-02	26258.6102	190.41
Gam190	5.6499E-03	2.0119E-05	3.7507E-04	1.5291E-02	18594.2158	268.90
Gam191	5.3477E-03	5.5532E-05	1.4105E-214	2.0262E-02	27848.3877	179.54
Gam192	1.1914E-02	1.2016E-04	4.8203E-04	4.0740E-02	42037.8420	118.94
Gam193	8.6947E-02	1.2275E-03	2.4052E-02	1.4001E-01	65323.2487	76.542
Gam194	4.3828E-03	1.4208E-05	5.8816E-05	1.2065E-02	24807.7751	201.54
Gam195	6.2229E-03	4.6592E-05	1.5081E-126	2.0269E-02	19968.8234	250.39
Gam196	3.9935E-03	1.6995E-05	5.8948E-06	1.1468E-02	22964.2384	217.72
Gam197	4.8131E-02	1.0506E-03	1.8951E-03	1.1265E-01	61968.1768	80.686
Gam198	3.8953E-02	4.6221E-04	4.3143E-03	7.6449E-02	103496.0248	48.311
Gam199	6.8304E-03	3.0269E-05	0.0000E+00	1.6100E-02	23752.2087	210.50
Gam200	3.8137E-03	1.2918E-05	0.0000E+00	1.1751E-02	29431.9507	169.88
Gam201	2.2061E-03	2.8632E-05	5.1399E-06	4.8039E-03	19079.3054	262.06
Gam202	3.9100E-03	1.0463E-05	1.0212E-05	1.0611E-02	42334.9988	118.10
Gam203	1.3941E-02	5.6850E-05	4.6010E-03	3.2667E-02	73870.7415	67.685
Gam204	8.8065E-03	4.3442E-05	2.1139E-04	1.6809E-02	25373.7124	197.05
Gam205	3.5903E-03	6.3083E-06	1.0064E-04	8.4176E-03	26858.5310	186.16
Gam206	1.8215E-03	3.9859E-06	0.0000E+00	5.5891E-03	21877.8619	228.54
Gam207	2.7125E-02	5.0313E-05	1.2766E-02	3.8572E-02	18578.7242	269.12
Gam208	4.5804E-03	1.6699E-05	1.4838E-05	1.1127E-02	36375.1408	137.45
Gam209	8.0613E-01	1.7607E-03	7.3269E-01	8.7467E-01	124530.5793	40.150
Gam210	8.4669E-03	1.9079E-05	2.2761E-03	1.5011E-02	20560.2395	243.18
Gam211	2.4497E-03	1.0519E-05	0.0000E+00	9.1696E-03	13154.7921	380.08
Gam212	2.7867E-02	8.6381E-05	1.1267E-02	4.2506E-02	27908.3662	179.15
Gam213	1.4054E-03	2.2605E-06	1.4660E-24	4.9417E-03	18466.9677	270.75
Gam214	1.2470E-02	4.7618E-05	1.3578E-03	2.4040E-02	57251.1230	87.334
Gam215	1.9740E-03	3.5207E-06	2.6479E-05	5.2427E-03	13247.3981	377.43
Gam216	2.5117E-03	2.6384E-05	2.9458E-05	6.2596E-03	19544.4539	255.82
Gam217	2.2047E-03	3.0671E-06	2.7441E-05	5.7185E-03	21443.2562	233.17
Gam218	2.1137E-02	1.0274E-04	3.6511E-03	3.9757E-02	32932.5392	151.82
Gam219	2.2999E-03	1.0651E-05	4.5831E-06	9.0745E-03	42058.9521	118.88
Gam220	2.8658E-03	2.5555E-05	1.5123E-05	9.6544E-03	22096.7836	226.27
Gam221	1.2031E-03	1.5336E-06	1.4189E-05	4.1287E-03	19430.3636	257.32
Gam222	3.4537E-02	5.9361E-04	1.6772E-04	7.8791E-02	189963.0833	26.320
Gam223	3.0790E-03	7.7608E-06	1.2559E-05	7.7606E-03	20839.1635	239.93

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
Gam224	1.6715E-03	1.8892E-06	4.5916E-05	4.3769E-03	15313.2087	326.51
Gam225	1.7481E-03	7.3093E-06	2.5438E-159	5.0969E-03	12502.8650	399.90
Gam226	3.6707E-02	9.9152E-04	3.0241E-04	1.1157E-01	47271.4019	105.77
Gam227	6.2283E-02	3.8712E-03	2.9392E-05	2.0028E-01	94429.1491	52.949
Gam228	2.7096E-02	2.2985E-04	4.3516E-03	5.4892E-02	26164.9348	191.09
Gam229	3.2028E-02	1.4356E-04	1.2579E-02	5.5498E-02	29452.0847	169.76
Gam230	1.0139E-02	5.1313E-05	4.9129E-04	2.3844E-02	65162.2063	76.731
Gam231	2.1222E-02	2.0195E-04	3.6462E-03	4.7430E-02	22587.0898	221.36
Gam232	3.0705E-02	1.8114E-04	8.2274E-03	5.9118E-02	36407.2808	137.33
Gam233	1.5053E-02	3.4100E-04	0.0000E+00	4.8016E-02	80339.2098	62.236
Gam234	3.7322E-02	1.6725E-04	1.0257E-02	5.9128E-02	12156.2454	411.31
Gam235	2.1579E-01	3.2940E-03	1.0663E-01	3.2862E-01	73331.2251	68.183
Gam236	1.2151E-02	9.8877E-05	8.8663E-05	3.5255E-02	23871.2828	209.45
Gam237	1.8690E-02	7.5997E-05	3.7458E-03	3.5588E-02	13149.2648	380.24
Gam238	6.3248E-03	4.0885E-05	2.4533E-07	2.1520E-02	31778.5257	157.33
Gam239	4.9375E-02	6.0725E-04	1.1591E-02	9.9927E-02	45728.0585	109.34
Gam240	6.7224E-03	4.2430E-05	0.0000E+00	1.8560E-02	36849.3569	135.68
Gam241	9.4317E-02	6.8467E-04	5.8860E-02	1.4398E-01	31663.8131	157.90
Gam242	1.7808E-02	2.4746E-04	3.8456E-06	5.3202E-02	45864.0363	109.01
Gam243	1.5343E-02	1.2346E-04	1.0887E-03	3.9775E-02	25997.7812	192.32
Gam244	5.1805E-02	2.3883E-04	2.8589E-02	8.9097E-02	14533.6772	344.02
Gam245	2.0058E-02	3.1468E-04	1.1706E-04	5.0474E-02	42293.9325	118.22
Gam246	7.4123E-02	5.8961E-04	3.8454E-02	1.3592E-01	35161.7526	142.20
Gam247	1.9739E-02	2.0427E-04	1.2378E-04	4.3046E-02	18502.4938	270.23
Gam248	4.6480E-02	7.2386E-04	1.1234E-02	1.0265E-01	31479.8738	158.83
Gam249	1.9907E-02	1.7395E-04	1.2233E-03	4.3735E-02	38284.4223	130.60
Gam250	5.8817E-02	3.0505E-04	2.9232E-02	8.8807E-02	48014.8455	104.13
Gam251	4.8917E-02	9.7887E-04	1.9972E-03	1.0607E-01	36523.3956	136.89
Gam252	6.4240E-03	5.2194E-05	8.1154E-06	2.3883E-02	33883.8243	147.56
Gam253	8.1575E-03	2.8191E-05	2.8640E-04	1.7617E-02	15709.6296	318.27
Gam254	5.3155E-03	2.9752E-05	0.0000E+00	1.2059E-02	28570.1134	175.00
Gam255	4.4334E-03	1.9147E-05	0.0000E+00	1.4631E-02	34892.3956	143.29
Gam256	5.4825E-03	2.9670E-05	1.3524E-04	1.4669E-02	19045.1706	262.53
Gam257	6.8300E-03	4.7635E-05	0.0000E+00	1.8941E-02	29033.1484	172.21
Gam258	1.3275E-02	9.1295E-05	7.4201E-04	3.0903E-02	28043.2345	178.29
Gam259	3.8815E-03	1.7372E-05	0.0000E+00	1.2537E-02	24454.0425	204.46
Gam260	1.0610E-02	5.7112E-05	3.7106E-04	2.4722E-02	31783.3337	157.31
Gam261	6.3711E-01	2.6489E-03	5.4235E-01	7.2877E-01	65788.9660	76.000
Gam262	8.8268E-03	9.8065E-05	3.1832E-05	2.7775E-02	24850.7622	201.20
Gam263	5.9210E-03	1.9170E-05	8.2042E-05	1.4713E-02	20750.6234	240.95
Gam264	7.3070E-02	3.7953E-04	4.4710E-02	1.1973E-01	40442.9323	123.63
Gam265	4.1779E-03	2.8985E-05	2.9634E-06	1.1795E-02	27245.2398	183.51
Gam266	3.6660E-03	1.0923E-05	0.0000E+00	1.1055E-02	28282.6364	176.78
Gam267	8.0333E-02	5.5329E-04	3.7376E-02	1.2613E-01	24671.3012	202.66
Gam268	3.5894E-02	5.3182E-04	9.1184E-04	7.6760E-02	22406.7080	223.14
Gam269	5.0000E-03	1.6646E-05	8.0503E-06	1.2885E-02	20377.3673	245.37
Gam270	6.0987E-03	5.0104E-05	0.0000E+00	1.9194E-02	20392.5245	245.18
Gam271	5.9221E-03	4.8925E-05	2.2163E-06	1.9486E-02	26498.4931	188.69
Gam272	5.4337E-03	3.5837E-05	3.2224E-05	1.7387E-02	24207.3908	206.54
Gam273	5.5041E-03	2.6215E-05	0.0000E+00	1.5638E-02	25953.9721	192.64
Gam274	5.7329E-03	1.9426E-05	6.6190E-06	1.4090E-02	13558.9442	368.76
Gam275	3.9834E-03	1.6668E-05	8.5724E-05	1.2867E-02	33148.8933	150.83
Gam276	2.9681E-03	7.5712E-06	3.9389E-05	7.4899E-03	24184.5647	206.74
Gam277	3.7934E-03	2.5369E-05	7.5299E-06	1.0718E-02	21527.6928	232.25

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam278	4.1185E-03	1.0647E-05	0.0000E+00	1.0465E-02	33540.4810	149.07
Gam279	4.2983E-03	1.5452E-05	3.3467E-06	1.0948E-02	25237.1937	198.12
Gam280	1.0858E-02	2.4034E-05	9.7111E-04	2.0302E-02	13321.8237	375.32
Gam281	5.0261E-03	8.7573E-06	1.0469E-03	1.1525E-02	13832.8982	361.45
Gam282	6.5803E-03	1.8902E-05	2.3867E-05	1.5060E-02	19289.4499	259.20
Gam283	5.0972E-03	2.5601E-05	3.0547E-05	1.6494E-02	20443.3423	244.57
Gam284	4.7241E-02	9.0826E-05	2.9230E-02	6.4696E-02	12236.9987	408.59
Gam285	5.9855E-03	1.7391E-05	1.7215E-05	1.3667E-02	16319.6658	306.37
Gam286	6.7989E-03	4.6773E-05	4.4576E-06	1.2852E-02	37887.2385	131.97
Gam287	7.0529E-01	6.5625E-04	6.5503E-01	7.5514E-01	69707.2689	71.728
Gam288	4.1422E-03	1.0170E-05	1.0113E-04	1.0852E-02	19079.0956	262.06
Gam289	4.8746E-02	1.5836E-04	3.0056E-02	7.2797E-02	26323.2367	189.94
Gam290	2.5543E-03	2.4406E-05	2.8525E-05	9.1834E-03	17013.3777	293.88
Gam291	1.1412E-03	3.3345E-06	0.0000E+00	2.4611E-03	17679.4058	282.81
Gam292	2.8599E-03	6.6291E-06	1.2848E-04	8.7528E-03	22401.1303	223.20
Gam293	1.1679E-01	2.8930E-04	8.4514E-02	1.4702E-01	67755.4724	73.794
Gam294	1.9097E-03	3.0563E-06	0.0000E+00	4.9579E-03	24372.6881	205.14
Gam295	2.5202E-03	6.4498E-06	0.0000E+00	6.4033E-03	38714.7146	129.14
Gam296	1.5994E-03	7.4314E-06	0.0000E+00	4.0390E-03	23306.1716	214.53
Gam297	3.7095E-03	6.1932E-05	7.2249E-05	1.0261E-02	17278.2065	289.38
Gam298	2.3703E-03	6.8958E-06	0.0000E+00	6.7324E-03	21231.4268	235.5000
Gam299	1.8840E-03	6.6609E-06	1.8302E-06	6.1451E-03	19788.1064	252.6770
Gam300	1.7226E-03	3.0834E-06	7.0403E-06	5.5546E-03	44347.4723	112.7460
Gam301	2.9327E-02	4.2531E-04	3.2592E-03	6.3432E-02	30510.9965	163.8753
Gam302	1.2290E-01	6.3671E-04	7.8302E-02	1.7072E-01	24623.5504	203.0576
Gam303	7.1168E-03	2.6278E-05	2.5334E-04	1.8564E-02	15769.2543	317.0727
Gam304	5.7247E-03	3.1242E-05	5.5859E-05	1.8575E-02	27674.4285	180.6722
Gam305	4.6415E-03	6.6660E-05	1.1610E-04	1.1787E-02	18508.2806	270.1494
Gam306	3.0414E-03	7.5922E-06	1.2860E-05	8.4326E-03	18594.0679	268.9030
Gam307	3.2661E-03	1.1904E-05	2.5661E-06	1.0671E-02	61553.7933	81.2298
Gam308	3.6162E-03	9.8520E-06	5.5995E-06	9.4542E-03	29464.3761	169.6964
Gam309	1.8790E-03	3.4469E-06	0.0000E+00	5.8284E-03	28845.1062	173.3396
Gam310	4.3399E-03	2.5936E-05	3.4381E-06	1.1586E-02	17978.2400	278.1140
Gam311	4.5431E-03	9.5163E-06	2.4009E-04	9.5250E-03	28937.6912	172.7850
Gam312	1.5384E-03	2.1657E-06	1.7842E-06	4.4478E-03	25970.4808	192.5263
Gam313	7.3209E-01	1.1326E-03	6.6362E-01	7.8946E-01	16866.8588	296.4393
Gam314	1.8622E-03	3.4278E-06	6.9320E-07	5.5181E-03	36575.9053	136.7020
Gam315	3.1327E-03	4.8260E-06	1.2577E-06	7.2462E-03	16861.5038	296.5335
Gam316	2.5437E-03	2.2604E-05	3.6101E-05	7.5467E-03	15749.6754	317.4669
Gam317	9.1956E-03	3.5181E-05	4.3308E-05	2.1015E-02	27536.5460	181.5769
Gam318	5.6658E-03	2.1964E-05	5.5923E-05	1.5150E-02	16059.1005	311.3499
Gam319	4.1368E-03	2.9159E-05	5.8963E-05	1.1556E-02	21005.5369	238.0325
Gam320	5.5329E-03	1.8947E-05	1.9608E-04	1.5499E-02	16956.3464	294.8748
Gam321	2.4158E-02	6.8231E-05	1.0261E-02	3.7861E-02	6766.1119	738.9768
Gam322	3.6947E-03	1.1449E-05	6.7122E-138	9.9310E-03	20442.0729	244.5936
Gam323	9.9583E-03	5.0129E-05	5.4610E-04	2.2615E-02	24769.9109	201.8578
Gam324	2.0962E-03	5.0383E-06	0.0000E+00	6.3166E-03	18788.8331	266.1155
Gam325	4.0082E-03	1.7494E-05	1.9199E-05	1.4044E-02	35960.8150	139.0402
Gam326	2.4669E-03	6.8498E-06	2.6688E-06	7.8518E-03	30845.1923	162.0998
Gam327	2.7149E-03	1.0769E-05	2.8700E-246	7.3696E-03	20326.7589	245.9812
Gam328	4.2796E-03	1.2370E-05	9.5174E-05	1.0872E-02	32612.1957	153.3169
Gam329	2.5964E-03	7.0825E-06	5.3002E-06	7.6621E-03	20614.7673	242.5446
Gam330	1.7047E-03	1.7693E-05	9.7357E-06	5.5125E-03	23923.4003	209.0004
Gam331	1.0921E-03	2.7578E-06	5.4262E-06	3.8869E-03	23087.1659	216.5705

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam332	1.5254E-03	4.0615E-06	2.2535E-05	3.9127E-03	8463.8927	590.7447
Gam333	4.5659E-03	9.2828E-06	1.0744E-04	1.0630E-02	43707.7232	114.3963
Gam334	8.1321E-03	2.3897E-05	1.5494E-03	1.8872E-02	22710.8803	220.1588
Gam335	1.6005E-02	4.9995E-05	5.6207E-03	2.8615E-02	64528.7687	77.4848
Gam336	2.3665E-02	6.6610E-05	1.0211E-02	4.3433E-02	21460.8618	232.9823
Gam337	4.5288E-02	9.5400E-05	2.6571E-02	6.3111E-02	35760.0416	139.8209
Gam338	1.9542E-03	2.6936E-06	1.1975E-105	5.3739E-03	16936.6509	295.2178
Gam339	7.3149E-01	6.6108E-04	6.8932E-01	7.8483E-01	41576.3816	120.2606
Gam340	1.4309E-03	1.5812E-05	8.7102E-06	3.1581E-03	17139.3611	291.7262
Gam341	1.5732E-03	4.4854E-06	0.0000E+00	4.3259E-03	24026.0963	208.1070
Gam342	5.0500E-03	1.5580E-05	2.3646E-04	1.1701E-02	33677.8372	148.4656
Gam343	1.2851E-01	3.7326E-04	9.3754E-02	1.6243E-01	50482.0442	99.0451
Gam344	1.1096E-03	1.0636E-05	1.0640E-103	3.2372E-03	20442.1127	244.5931
Gam345	1.1524E-03	1.5336E-06	2.3512E-05	3.4638E-03	15780.5214	316.8463
Gam346	2.6502E-03	7.4835E-06	1.1649E-209	6.1499E-03	23170.2820	215.7937
Gam347	2.4587E-03	1.0587E-05	2.4929E-66	7.9508E-03	26962.0248	185.4460
Gam348	2.5180E-03	4.8702E-06	0.0000E+00	7.5496E-03	18700.3745	267.3743
Gam349	2.8176E-03	9.8475E-06	5.2767E-05	8.0141E-03	20869.7338	239.5814
Gam350	3.2411E-03	2.9135E-05	2.8385E-06	1.0693E-02	25938.5472	192.7633
Gam351	2.4793E-02	3.3018E-04	9.0062E-04	5.9350E-02	33327.8158	150.0248
Gam352	5.1001E-02	6.1874E-04	2.6070E-03	9.2298E-02	37286.2379	134.0977
Gam353	6.1075E-03	6.7308E-05	0.0000E+00	2.5894E-02	38588.0048	129.5739
Gam354	3.7852E-03	1.4613E-05	0.0000E+00	1.2001E-02	8916.7918	560.7398
Gam355	7.7790E-03	6.9994E-05	2.7015E-05	2.5643E-02	31182.7687	160.3450
Gam356	4.9178E-03	3.2680E-05	0.0000E+00	1.7044E-02	34944.3649	143.0846
Gam357	6.1894E-03	5.0146E-05	0.0000E+00	1.8185E-02	29584.5974	169.0069
Gam358	7.1705E-03	6.8995E-05	0.0000E+00	2.7270E-02	24876.9587	200.9892
Gam359	4.6590E-03	2.1364E-05	6.4461E-06	1.3523E-02	28535.3499	175.2213
Gam360	9.1733E-03	5.5219E-05	3.8856E-05	2.3279E-02	37390.2793	133.7246
Gam361	6.2608E-03	7.4079E-05	0.0000E+00	2.5291E-02	40482.6649	123.5097
Gam362	3.8615E-03	1.0624E-05	7.8225E-06	9.8337E-03	14735.0059	339.3280
Gam363	7.0250E-03	5.3764E-05	0.0000E+00	2.0566E-02	24005.2184	208.2880
Gam364	4.3183E-03	2.7370E-05	1.8785E-06	1.2305E-02	29506.0729	169.4566
Gam365	6.0414E-01	3.5649E-03	4.6988E-01	7.0603E-01	45696.0153	109.4187
Gam366	1.6645E-02	1.3086E-04	4.1311E-04	3.2876E-02	12823.9967	389.8940
Gam367	3.3010E-02	3.7879E-04	6.9911E-03	6.6756E-02	28993.9751	172.4496
Gam368	1.8315E-02	1.6982E-04	3.3724E-04	4.5120E-02	12876.7647	388.2963
Gam369	6.8801E-03	4.4235E-05	1.1847E-04	2.0388E-02	28752.5357	173.8977
Gam370	1.0269E-01	1.5614E-03	2.4741E-02	1.7153E-01	36482.7202	137.0512
Gam371	1.8484E-02	1.1383E-04	1.8662E-04	3.6468E-02	22382.4058	223.3897
Gam372	1.0651E-02	1.2899E-04	0.0000E+00	3.1775E-02	20097.0589	248.7926
Gam373	1.7435E-02	1.5667E-04	3.3653E-04	3.8187E-02	20262.4381	246.7620
Gam374	8.3769E-03	9.3021E-05	0.0000E+00	3.0907E-02	36945.3840	135.3349
Gam375	1.6333E-02	1.4734E-04	8.4483E-04	4.1077E-02	27353.5296	182.7918
Gam376	2.5206E-03	5.0930E-06	2.9205E-05	7.2401E-03	20979.2645	238.3306
Gam377	3.1381E-03	9.6389E-06	0.0000E+00	9.6390E-03	23175.4021	215.7460
Gam378	3.2783E-03	4.6262E-06	1.4914E-04	7.0207E-03	18371.5781	272.1595
Gam379	1.9466E-03	4.3353E-06	1.9932E-05	6.0063E-03	37457.2433	133.4855
Gam380	1.5848E-03	3.9237E-06	0.0000E+00	4.5166E-03	20140.3158	248.2583
Gam381	3.8456E-03	2.2601E-05	2.3276E-05	1.4362E-02	28454.7058	175.7179
Gam382	2.3889E-03	1.7898E-05	5.8433E-06	7.2489E-03	20698.5296	241.5631
Gam383	1.9750E-03	4.4970E-06	5.0407E-06	5.5307E-03	20741.2644	241.0653
Gam384	3.1523E-03	6.6443E-06	1.1953E-04	7.5092E-03	20241.7043	247.0148
Gam385	3.1986E-02	6.3682E-05	1.9275E-02	4.8261E-02	20170.9672	247.8810

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam386	1.9295E-03	5.7482E-06	9.4874E-06	6.3732E-03	21484.8529	232.7221
Gam387	1.3705E-03	2.1416E-06	9.1667E-06	4.3620E-03	30328.2155	164.8630
Gam388	1.6270E-03	2.5758E-06	1.9328E-07	4.6204E-03	27632.8669	180.9439
Gam389	1.7055E-03	2.6181E-06	0.0000E+00	5.0687E-03	27168.1551	184.0390
Gam390	1.0854E-02	2.1510E-05	3.3281E-03	2.0403E-02	29059.2892	172.0620
Gam391	7.1509E-01	5.9987E-04	6.6839E-01	7.6232E-01	16853.6152	296.6723
Gam392	2.8702E-03	1.3946E-05	0.0000E+00	1.1007E-02	33926.4352	147.3777
Gam393	3.2691E-03	1.4818E-05	5.2630E-06	1.2622E-02	12896.3590	387.7063
Gam394	3.4814E-02	9.0375E-05	1.6997E-02	5.2017E-02	22320.0818	224.0135
Gam395	1.3375E-01	2.6785E-04	9.8756E-02	1.6443E-01	31435.8188	159.0542
Gam396	3.6185E-03	1.2274E-05	2.1532E-04	1.0387E-02	17364.7334	287.9399
Gam397	3.5225E-03	1.0651E-05	0.0000E+00	9.8298E-03	23028.6780	217.1206
Gam398	1.2023E-02	4.8850E-05	5.7406E-04	2.4273E-02	43936.2306	113.8013
Gam399	7.1335E-03	2.2365E-05	2.8489E-04	1.5523E-02	19134.3241	261.3105
Gam400	1.0605E-02	3.5265E-05	2.2637E-03	1.9447E-02	18513.1736	270.0780
Gam401	1.4145E-01	5.6709E-04	9.5771E-02	1.8805E-01	46561.6260	107.3846
Gam402	1.1253E-02	6.0266E-05	8.2796E-06	2.6947E-02	31745.4089	157.5031
Gam403	1.0800E-02	2.5388E-05	1.8994E-03	2.0844E-02	14008.5219	356.9256
Gam404	1.4723E-03	2.4668E-06	8.5575E-07	4.0427E-03	27293.1690	183.1960
Gam405	5.1690E-03	6.1864E-06	8.0014E-04	1.0367E-02	17573.1706	284.5246
Gam406	2.4912E-03	4.0882E-06	1.5439E-04	6.6381E-03	17474.4905	286.1314
Gam407	2.6837E-03	5.2157E-06	3.7603E-05	7.0031E-03	25231.3303	198.1663
Gam408	2.7472E-03	7.4211E-06	4.1349E-05	8.4109E-03	83638.5474	59.7810
Gam409	2.4838E-03	8.3758E-06	3.0316E-05	8.3449E-03	14372.5505	347.8854
Gam410	9.7625E-03	3.1184E-05	8.8820E-04	2.0287E-02	23276.3846	214.8100
Gam411	2.5801E-02	7.9244E-05	6.7707E-03	4.3522E-02	31942.0713	156.5334
Gam412	3.0284E-03	6.9727E-06	1.9368E-05	7.8618E-03	24849.1321	201.2143
Gam413	4.9011E-03	1.0844E-05	9.8124E-05	1.0737E-02	12502.6231	399.9161
Gam414	6.3975E-03	1.7203E-05	1.3161E-03	1.6126E-02	13817.0982	361.8705
Gam415	5.2195E-03	1.0166E-05	1.4682E-04	1.1186E-02	17327.3056	288.5619
Gam416	2.4188E-03	3.8366E-06	0.0000E+00	5.6751E-03	13496.7349	370.4600
Gam417	7.0950E-01	6.3763E-04	6.5662E-01	7.5761E-01	46169.2866	108.2971
Gam418	1.0090E-02	4.1201E-05	2.1459E-04	1.9922E-02	49123.0598	101.7852
Gam419	2.8005E-03	7.3404E-06	4.7947E-05	9.7403E-03	25931.2775	192.8173
Gam420	4.8523E-03	2.1230E-05	1.3610E-05	1.5738E-02	16063.6154	311.2624
Gam421	2.1635E-02	6.2529E-05	6.5801E-03	3.7325E-02	16900.7761	295.8444
Gam422	2.8975E-03	6.7063E-06	0.0000E+00	8.1080E-03	27776.0669	180.0111
Gam423	5.6688E-03	2.0444E-05	6.2900E-05	1.4027E-02	15309.3126	326.5986
Gam424	2.8593E-03	8.1163E-06	8.5713E-06	8.6236E-03	64005.0544	78.1188
Gam425	1.6124E-03	2.2344E-06	3.4202E-06	4.5395E-03	25226.5574	198.2038
Gam426	2.0740E-03	4.0899E-06	2.0857E-05	6.1500E-03	31370.0390	159.3878
Gam427	9.9544E-03	2.0338E-05	6.3560E-04	1.6414E-02	43002.3537	116.2727
Gam428	2.9525E-03	7.1782E-06	2.1226E-04	7.0641E-03	25471.4357	196.2983
Gam429	1.6369E-03	2.7502E-06	9.2684E-06	4.9787E-03	24336.0265	205.4567
Gam430	1.3922E-03	3.1853E-06	0.0000E+00	5.0678E-03	32406.8350	154.2884
Gam431	1.4245E-03	6.8981E-06	5.7031E-06	4.4679E-03	26461.6035	188.9530
Gam432	1.4795E-03	2.3451E-06	6.7581E-06	3.8377E-03	28039.3865	178.3206
Gam433	1.5035E-02	6.3199E-05	3.4628E-03	2.6602E-02	43256.3488	115.5900
Gam434	5.0448E-03	7.4097E-06	3.6836E-04	1.0538E-02	33605.9976	148.7830
Gam435	1.7197E-03	9.7988E-06	6.9005E-06	4.5959E-03	14081.1043	355.0858
Gam436	3.7395E-03	1.7747E-05	9.4233E-05	1.1876E-02	17872.4177	279.7607
Gam437	4.4831E-02	8.2069E-05	2.4317E-02	5.9053E-02	55420.5429	90.2193
Gam438	2.0099E-03	2.9398E-06	4.0242E-06	5.3215E-03	23310.4666	214.4959
Gam439	5.7917E-02	5.4776E-05	4.4385E-02	7.1556E-02	18974.4135	263.5128

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam440	1.2071E-03	8.7368E-07	2.0884E-05	3.0730E-03	21556.4801	231.9488
Gam441	7.7877E-04	1.2320E-06	3.9643E-06	2.7132E-03	34442.7165	145.1686
Gam442	2.8372E-03	6.8045E-06	3.3803E-06	5.9055E-03	14566.5388	343.2524
Gam443	8.3304E-01	1.6314E-04	8.0840E-01	8.5844E-01	32451.6730	154.0753
Gam444	7.0302E-04	5.6680E-07	2.1415E-235	1.9467E-03	22644.6403	220.8028
Gam445	1.2805E-03	1.7479E-06	0.0000E+00	3.6397E-03	22109.1297	226.1509
Gam446	7.8547E-04	8.4742E-07	1.3043E-05	2.7585E-03	31651.8098	157.9689
Gam447	2.0870E-03	4.3782E-06	1.0134E-05	6.9000E-03	20642.3162	242.2209
Gam448	2.9245E-03	1.2113E-05	2.6652E-06	6.9217E-03	13529.3141	369.5679
Gam449	1.1445E-03	1.0144E-06	1.9544E-05	3.1386E-03	20352.2298	245.6733
Gam450	2.0047E-03	3.2108E-06	3.2630E-06	5.7242E-03	17725.7266	282.0759
Gam451	7.1105E-03	4.8820E-05	5.8517E-06	2.3692E-02	26338.7144	189.8346
Gam452	1.2013E-02	1.0252E-04	1.1897E-05	3.1603E-02	23952.7096	208.7447
Gam453	4.1419E-03	3.2262E-05	8.8946E-05	1.6086E-02	48457.2786	103.1837
Gam454	6.0563E-03	1.1015E-04	0.0000E+00	1.7585E-02	22774.3455	219.5453
Gam455	5.1958E-03	1.9394E-05	1.4232E-04	1.3593E-02	29107.8441	171.7750
Gam456	9.3003E-03	3.1841E-05	3.9921E-04	2.1125E-02	24715.7050	202.3005
Gam457	5.6489E-03	2.3863E-05	0.0000E+00	1.6950E-02	25998.2037	192.3210
Gam458	2.2142E-03	5.1309E-06	0.0000E+00	7.4352E-03	33010.7600	151.4658
Gam459	3.2802E-03	1.5509E-05	0.0000E+00	8.8765E-03	11562.4319	432.4350
Gam460	2.5709E-02	1.0707E-04	7.3453E-03	4.4883E-02	19696.9727	253.8461
Gam461	3.2744E-03	8.1703E-06	2.0800E-05	8.9429E-03	50279.2180	99.4447
Gam462	3.7221E-03	1.0454E-05	2.1471E-06	1.1410E-02	21433.3031	233.2818
Gam463	3.6508E-03	1.4389E-05	1.4977E-05	1.0892E-02	28069.2042	178.1312
Gam464	4.3139E-03	1.7651E-05	5.4271E-05	1.2524E-02	21386.6295	233.7909
Gam465	6.8082E-03	2.5981E-05	3.0086E-04	1.5746E-02	15262.2073	327.6066
Gam466	6.6788E-02	2.5808E-04	3.8270E-02	9.6085E-02	33162.6412	150.7721
Gam467	4.7363E-03	2.5146E-05	2.1756E-05	1.3580E-02	24894.7832	200.8453
Gam468	2.5679E-03	1.2170E-05	0.0000E+00	9.7807E-03	45902.5130	108.9265
Gam469	5.5808E-01	1.4357E-03	4.7133E-01	6.2205E-01	28278.1542	176.8149
Gam470	1.6805E-01	8.0602E-04	1.2388E-01	2.3427E-01	35056.1824	142.6282
Gam471	4.1599E-03	2.0401E-05	1.7421E-06	1.2528E-02	17364.7550	287.9396
Gam472	7.6984E-03	5.7707E-05	1.7294E-66	2.1739E-02	33902.9657	147.4797
Gam473	5.9583E-02	5.5918E-04	1.4012E-02	9.9320E-02	102335.1947	48.8590
Gam474	5.1308E-03	3.7137E-05	2.9308E-05	1.9383E-02	18410.6335	271.5822
Gam475	2.0768E-02	6.8060E-05	6.6840E-03	3.7132E-02	18835.5876	265.4550
Gam476	4.5246E-03	2.9527E-05	1.2246E-05	1.5367E-02	28418.4006	175.9423
Gam477	7.9723E-03	3.4245E-05	3.6387E-04	1.6831E-02	28131.9693	177.7337
Gam478	1.8001E-03	3.8621E-06	3.6605E-06	5.4703E-03	34500.3679	144.9260
Gam479	2.1294E-03	2.6664E-06	2.6298E-06	5.2907E-03	10274.7777	486.6285
Gam480	1.8446E-03	3.9387E-06	2.2398E-05	5.2873E-03	45227.5387	110.5521
Gam481	1.7509E-03	3.8855E-06	3.6611E-97	4.4719E-03	21426.9780	233.3507
Gam482	2.7513E-03	1.2285E-05	1.4434E-05	6.9678E-03	25084.5453	199.3259
Gam483	3.2090E-03	9.4208E-06	6.0894E-05	8.6159E-03	35241.6479	141.8776
Gam484	1.2637E-03	1.5646E-06	0.0000E+00	3.2678E-03	26065.9621	191.8210
Gam485	6.6660E-03	1.2467E-05	1.1263E-03	1.3715E-02	21821.2805	229.1341
Gam486	9.9720E-04	1.2262E-06	0.0000E+00	3.8547E-03	40447.7451	123.6163
Gam487	1.6581E-03	5.8962E-06	8.4528E-06	5.3982E-03	11862.0810	421.5112
Gam488	3.1126E-03	6.7946E-06	6.7868E-05	7.3754E-03	23350.4316	214.1288
Gam489	1.2443E-03	2.4282E-06	6.0437E-06	3.9047E-03	26562.3243	188.2365
Gam490	1.1049E-02	2.9569E-05	8.7339E-04	1.9780E-02	23895.0789	209.2481
Gam491	4.8344E-02	1.0015E-04	2.9564E-02	6.9302E-02	25991.7975	192.3684
Gam492	2.3050E-03	3.3382E-06	9.9957E-05	5.4752E-03	17442.8134	286.6510
Gam493	1.8940E-03	1.0811E-05	0.0000E+00	4.7140E-03	30163.3613	165.7640

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam494	3.2402E-02	5.7514E-05	2.1006E-02	4.7081E-02	24356.8705	205.2809
Gam495	8.2957E-01	3.3422E-04	7.9682E-01	8.6667E-01	20813.6358	240.2271
Gam496	1.7734E-03	2.7448E-06	1.5778E-06	5.3464E-03	27772.7873	180.0323
Gam497	2.8291E-03	7.4174E-06	2.0672E-238	7.7762E-03	66433.4908	75.2632
Gam498	1.9807E-02	1.1193E-04	4.7007E-03	3.9870E-02	55844.1523	89.5349
Gam499	3.7384E-03	1.0750E-05	6.1443E-06	9.4147E-03	27539.7138	181.5560
Gam500	5.3652E-03	9.2246E-06	8.6062E-04	1.1890E-02	72948.9489	68.5411
Gam501	1.3157E-01	2.7214E-04	1.0350E-01	1.6442E-01	16829.9330	297.0897
Gam502	3.6686E-03	1.1225E-05	1.7460E-06	1.0535E-02	37864.7786	132.0488
Gam503	8.1465E-03	1.5319E-05	2.2713E-03	1.5033E-02	16359.8070	305.6271
Gam504	3.0005E-03	7.1040E-06	2.8331E-07	8.7553E-03	22353.0670	223.6830
Gam505	2.3504E-03	4.2734E-06	6.3767E-06	6.3886E-03	19613.4301	254.9274
Gam506	3.4265E-03	6.4921E-06	1.4338E-04	7.8648E-03	28198.0034	177.3175
Gam507	2.2086E-03	4.2489E-06	6.6654E-06	6.9498E-03	28960.2604	172.6504
Gam508	1.8329E-03	3.4288E-06	2.5296E-05	4.4587E-03	20578.9871	242.9663
Gam509	1.7202E-03	3.1004E-06	0.0000E+00	5.2366E-03	22627.6876	220.9682
Gam510	2.4846E-02	5.0564E-05	8.7965E-03	3.6573E-02	19652.0538	254.4263
Gam511	3.0427E-03	7.0590E-06	4.7212E-07	7.9829E-03	24954.0345	200.3684
Gam512	1.6581E-03	2.4683E-06	5.2152E-07	4.7452E-03	42106.3515	118.7469
Gam513	2.7028E-03	7.0896E-06	3.5590E-205	8.0023E-03	33245.8452	150.3947
Gam514	2.5833E-03	8.3656E-06	1.1496E-06	6.5102E-03	21024.7301	237.8152
Gam515	1.4949E-02	2.1112E-05	7.6734E-03	2.3874E-02	14787.2077	338.1301
Gam516	4.5785E-03	9.8908E-06	1.4013E-04	9.7391E-03	28731.1410	174.0272
Gam517	2.1947E-02	1.2392E-04	7.0573E-03	4.2178E-02	11922.9571	419.3591
Gam518	2.0214E-03	3.1687E-06	1.8265E-05	5.6039E-03	16504.4170	302.9492
Gam519	1.9063E-03	1.5384E-05	0.0000E+00	5.5721E-03	17525.8421	285.2930
Gam520	2.5680E-03	1.0041E-05	1.2352E-253	7.7990E-03	17917.6273	279.0548
Gam521	7.4956E-01	6.0872E-04	7.0241E-01	7.8861E-01	30806.1306	162.3054
Gam522	2.2303E-03	4.3630E-06	1.5289E-05	5.6639E-03	9786.4206	510.9120
Gam523	3.3383E-03	2.8680E-05	1.7586E-06	1.4263E-02	31699.7803	157.7298
Gam524	1.2529E-03	1.4225E-06	0.0000E+00	3.6943E-03	12450.4991	401.5903
Gam525	2.8892E-03	3.2576E-05	2.7673E-06	1.0769E-02	24567.9091	203.5175
Gam526	1.0670E-03	1.3093E-06	3.7588E-305	3.7336E-03	25525.8323	195.8800
Gam527	6.8261E-03	2.3397E-05	2.8033E-04	1.6208E-02	51743.5942	96.6303
Gam528	3.3637E-02	2.6144E-04	2.8814E-03	5.1314E-02	56874.1537	87.9134
Gam529	1.2611E-02	3.3020E-05	1.6657E-03	2.2728E-02	169947.5985	29.4208
Gam530	1.6999E-03	5.2823E-06	1.3537E-05	4.3540E-03	25726.4129	194.3528
Gam531	3.5967E-03	3.5060E-06	8.3529E-06	7.0724E-03	17345.1166	288.2656
Gam532	3.1536E-03	5.6114E-06	7.2577E-113	7.7095E-03	33188.7328	150.6535
Gam533	8.3641E-03	1.9661E-05	1.6485E-03	1.7416E-02	26295.3614	190.1476
Gam534	8.4514E-03	2.7209E-05	2.8089E-04	1.7232E-02	124216.1730	40.2524
Gam535	2.6716E-03	1.3608E-05	4.0078E-06	7.4991E-03	81553.5622	61.3094
Gam536	9.3193E-04	2.9252E-06	7.9932E-06	3.8403E-03	18179.8773	275.0294
Gam537	1.1114E-03	1.3498E-06	6.6574E-06	3.2932E-03	19534.9622	255.9514
Gam538	1.5258E-03	3.8594E-06	2.1416E-06	4.9796E-03	15821.6917	316.0218
Gam539	1.3697E-03	6.3098E-06	2.7123E-06	5.3163E-03	18226.7989	274.3213
Gam540	1.8537E-03	8.1875E-06	9.3136E-07	8.5293E-03	103611.6168	48.2571
Gam541	1.5786E-03	8.1487E-06	8.1707E-06	4.0892E-03	21485.0467	232.7200
Gam542	1.4865E-03	6.0516E-06	0.0000E+00	5.9249E-03	27535.5807	181.5832
Gam543	1.1454E-03	2.7441E-06	9.3162E-06	4.0491E-03	28161.6175	177.5466
Gam544	1.0501E-03	3.8784E-06	9.7212E-238	4.1070E-03	17075.4013	292.8189
Gam545	1.1498E-03	1.1439E-06	2.2421E-06	3.1272E-03	12510.7885	399.6551
Gam546	8.7337E-04	2.8368E-06	1.0350E-05	2.3897E-03	12978.9781	385.2383
Gam547	8.8294E-01	2.7952E-04	8.5074E-01	9.1435E-01	18172.1093	275.1469

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam548	5.6532E-03	2.3565E-05	6.5690E-05	1.6203E-02	123254.6503	40.5664
Gam549	1.4508E-03	1.4327E-05	6.4857E-07	4.0106E-03	10878.6212	459.6171
Gam550	1.3803E-02	1.8331E-04	2.2917E-03	4.4794E-02	34666.3023	144.2323
Gam551	6.7310E-03	2.7680E-05	1.1266E-04	1.7192E-02	35142.0623	142.2796
Gam552	2.4209E-01	8.7090E-03	3.8008E-02	3.7133E-01	162869.1094	30.6995
Gam553	3.1470E-03	6.9356E-06	2.8290E-05	8.0054E-03	33272.2618	150.2753
Gam554	3.9453E-03	1.3509E-05	1.0103E-04	1.2127E-02	24520.4763	203.9112
Gam555	3.6845E-03	9.3521E-06	5.2495E-05	9.5950E-03	24818.6934	201.4610
Gam556	7.3746E-03	2.0730E-05	1.2310E-03	1.6435E-02	29287.3525	170.7222
Gam557	4.5495E-03	1.0316E-05	3.0097E-04	1.0594E-02	21781.4116	229.5535
Gam558	9.9648E-03	5.5931E-05	9.4161E-04	2.7924E-02	32907.5859	151.9407
Gam559	1.9224E-03	3.7264E-06	1.4318E-05	5.3916E-03	25555.6939	195.6511
Gam560	8.1419E-03	2.9276E-05	1.1023E-03	1.9852E-02	81626.1404	61.2549
Gam561	1.9963E-03	7.0339E-06	1.4485E-286	5.6362E-03	39974.1572	125.0808
Gam562	1.4446E-03	2.2048E-06	1.4879E-05	4.5418E-03	44515.1380	112.3213
Gam563	6.0201E-03	5.3759E-05	9.1406E-05	1.8311E-02	43368.0244	115.2923
Gam564	2.6932E-03	6.5481E-06	8.6165E-05	7.5424E-03	19725.0931	253.4842
Gam565	4.9005E-03	8.1615E-06	3.5200E-04	1.0600E-02	45810.4813	109.1453
Gam566	6.7075E-03	6.6712E-05	3.4673E-06	1.4944E-02	50579.6758	98.8539
Gam567	3.6168E-03	1.1999E-05	1.0541E-04	1.0273E-02	22432.4382	222.8915
Gam568	5.4354E-03	2.0577E-05	4.0865E-07	1.4374E-02	34782.8385	143.7491
Gam569	1.3744E-02	4.3454E-05	4.8660E-03	2.7866E-02	89800.7279	55.6788
Gam570	2.0294E-02	2.9998E-04	2.1426E-03	6.0205E-02	108491.3389	46.0866
Gam571	1.2219E-03	2.3025E-06	0.0000E+00	3.5279E-03	26854.9602	186.1853
Gam572	1.0003E-02	3.9214E-05	6.9248E-04	2.2608E-02	70626.9782	70.7945
Gam573	6.1290E-01	1.9880E-02	3.9848E-01	9.0596E-01	63459.6414	78.7902
Gam574	7.6587E-03	2.6901E-05	1.7922E-04	1.6135E-02	37523.9088	133.2484
Gam575	9.8089E-03	4.2591E-05	1.0069E-03	2.3786E-02	63298.2454	78.9911
Gam576	8.3441E-03	5.9303E-05	1.0641E-04	2.4359E-02	48248.9802	103.6291
Gam577	1.1359E-01	5.5179E-04	7.9416E-02	1.6391E-01	29470.7144	169.6600
Gam578	2.0572E-03	4.9978E-06	1.9914E-06	5.1430E-03	19695.0448	253.8710
Gam579	5.0992E-03	1.8647E-05	1.0823E-05	1.2779E-02	45835.5124	109.0857
Gam580	3.0145E-03	1.6992E-05	9.0696E-06	9.2287E-03	18390.8363	271.8745
Gam581	6.3175E-03	2.1863E-05	2.0929E-04	1.5309E-02	37372.7549	133.7873
Gam582	5.5664E-03	1.5330E-05	2.5135E-04	1.2761E-02	27636.1311	180.9226
Gam583	7.1495E-03	2.5749E-05	2.6355E-05	1.7411E-02	16700.0796	299.3998
Gam584	2.3959E-03	9.8418E-06	0.0000E+00	8.6192E-03	31609.9241	158.1782
Gam585	1.3901E-02	5.7898E-05	1.1593E-04	2.6953E-02	28774.4422	173.7653
Gam586	4.5053E-03	1.5922E-05	1.3999E-05	1.1896E-02	24670.6474	202.6700
Gam587	2.3749E-03	4.3713E-06	4.6294E-06	6.3873E-03	18980.6039	263.4268
Gam588	6.2443E-03	1.9280E-05	6.4128E-04	1.6022E-02	19087.7909	261.9475
Gam589	2.6017E-03	7.3585E-06	0.0000E+00	7.6996E-03	29942.2713	166.9880
Gam590	3.9186E-03	3.8209E-05	8.5562E-07	8.2811E-03	38640.8552	129.3967
Gam591	3.3120E-02	1.3032E-04	1.4226E-02	5.1841E-02	14839.2761	336.9437
Gam592	1.1356E-02	6.4249E-05	3.9847E-04	2.5152E-02	36637.9856	136.4704
Gam593	2.5612E-03	7.2270E-06	1.8899E-05	8.2262E-03	43793.8830	114.1712
Gam594	4.0829E-03	9.9005E-06	1.9282E-139	9.1253E-03	26159.3926	191.1359
Gam595	1.6455E-02	1.4091E-04	9.8861E-04	4.0410E-02	25487.0538	196.1780
Gam596	4.6105E-03	7.1124E-05	1.6447E-05	2.3031E-02	12820.6463	389.9959
Gam597	1.2425E-02	1.0413E-04	2.9849E-04	3.5192E-02	29425.0675	169.9231
Gam598	5.9868E-03	2.9668E-05	9.7568E-307	1.4641E-02	19891.5709	251.3628
Gam599	7.1850E-01	1.4458E-03	6.4775E-01	7.9540E-01	43147.8720	115.8806
Gam600	3.8137E-03	1.5407E-05	3.7176E-107	1.2743E-02	36807.9025	135.8404
Gam601	4.5700E-03	2.5936E-05	1.2255E-05	1.4629E-02	61878.8181	80.8031

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
Gam602	5.9703E-02	5.4771E-04	8.3209E-03	9.8580E-02	97691.4403	51.1816
Gam603	3.7547E-03	1.2219E-05	1.3979E-06	1.0969E-02	18740.0983	266.8076
Gam604	3.3626E-03	3.0958E-05	7.2764E-06	9.5617E-03	23786.6937	210.2016
Gam605	2.8119E-03	1.2861E-05	1.2029E-05	1.0376E-02	39130.1276	127.7788
Gam606	8.8095E-03	3.7352E-05	7.7500E-04	2.0562E-02	16802.9664	297.5665
Gam607	5.9677E-03	2.1292E-05	1.5958E-05	1.4337E-02	26793.4490	186.6128
Gam608	6.4906E-03	3.8496E-05	8.2154E-05	1.8456E-02	25738.6885	194.2601
Gam609	3.4075E-03	1.1365E-05	1.9948E-06	9.8028E-03	19701.1370	253.7925
Gam610	3.7020E-02	1.5706E-04	1.8155E-02	6.0903E-02	30342.1325	164.7874
Gam611	2.7671E-03	7.2091E-06	1.7110E-06	8.7172E-03	13700.4153	364.9524
Gam612	2.5592E-03	7.2477E-06	0.0000E+00	7.1332E-03	42692.5417	117.1165
Gam613	5.1943E-03	1.5530E-05	1.3503E-06	1.2906E-02	51838.7161	96.4530
Gam614	3.7065E-03	1.0699E-05	2.1408E-06	1.0606E-02	24862.7780	201.1038
Gam615	4.0503E-03	9.8144E-06	4.5949E-05	1.0229E-02	42991.9561	116.3008
Gam616	1.5509E-02	6.6455E-05	5.7150E-03	3.5507E-02	34656.8095	144.2718
Gam617	5.9456E-03	2.8665E-05	1.7264E-04	1.7430E-02	31807.6331	157.1950
Gam618	1.0786E-02	5.9151E-05	5.5395E-05	2.5488E-02	16244.7560	307.7916
Gam619	8.3767E-03	2.0980E-05	3.4127E-04	1.7033E-02	16412.7446	304.6413
Gam620	1.8622E-02	2.2345E-04	2.2046E-04	5.1229E-02	134100.5341	37.2855
Gam621	3.7281E-03	1.7661E-05	0.0000E+00	1.0159E-02	30196.9492	165.5796
Gam622	9.0645E-02	2.0393E-03	2.7301E-02	1.7825E-01	13895.2991	359.8339
Gam623	1.4109E-02	1.2263E-04	8.9471E-05	3.6890E-02	39533.6827	126.4744
Gam625	6.7281E-01	1.8216E-03	5.8844E-01	7.4477E-01	102972.9902	48.5564
Gam624	5.2883E-03	2.2810E-05	0.0000E+00	1.3935E-02	33696.7899	148.3821
mean00	3.3153E+01	6.2187E-06	3.3148E+01	3.3157E+01	92575.8051	54.009
mean01	1.3207E+02	2.1123E-05	1.3206E+02	1.3207E+02	40296.2464	124.08
mean02	3.3530E+01	1.1900E-05	3.3524E+01	3.3535E+01	23306.5483	214.53
mean03	1.3223E+02	1.5145E-05	1.3223E+02	1.3224E+02	18904.7864	264.48
mean04	3.3143E+01	3.1642E-05	3.3134E+01	3.3154E+01	312025.9022	16.024
mean05	1.3222E+02	9.8978E-05	1.3221E+02	1.3224E+02	185982.0670	26.884
mean06	3.3227E+01	4.6647E-05	3.3215E+01	3.3240E+01	142592.6571	35.064
mean07	1.3218E+02	3.5168E-05	1.3217E+02	1.3219E+02	45558.0184	109.75
mean08	3.3646E+01	4.3518E-05	3.3640E+01	3.3660E+01	36341.5325	137.58
mean09	1.3244E+02	1.0831E-05	1.3244E+02	1.3245E+02	14098.1236	354.65
mean10	3.3349E+01	2.8868E-05	3.3337E+01	3.3358E+01	42864.7199	116.64
mean11	1.3212E+02	3.0243E-05	1.3211E+02	1.3213E+02	19599.8797	255.10
mean12	3.3524E+01	5.5641E-05	3.3512E+01	3.3540E+01	55988.4161	89.304
mean13	1.3238E+02	4.4698E-05	1.3237E+02	1.3239E+02	40444.0229	123.62
mean14	3.3468E+01	4.4562E-05	3.3455E+01	3.3476E+01	136234.5080	36.701
mean15	1.3274E+02	4.3066E-05	1.3274E+02	1.3276E+02	142630.1781	35.055
mean16	3.3375E+01	9.4142E-05	3.3356E+01	3.3389E+01	113550.4454	44.033
mean17	1.3251E+02	7.9729E-05	1.3249E+02	1.3252E+02	42178.8880	118.54
mean18	3.3736E+01	6.2646E-04	3.3699E+01	3.3785E+01	214245.2839	23.337
mean19	1.3315E+02	1.3344E-03	1.3309E+02	1.3322E+02	104416.6575	47.885
mean20	3.3720E+01	7.8763E-05	3.3704E+01	3.3737E+01	69508.4940	71.933
mean21	1.3294E+02	9.5506E-05	1.3292E+02	1.3296E+02	108031.1650	46.282
mean22	3.3536E+01	5.8907E-06	3.3531E+01	3.3541E+01	18076.7866	276.59
mean23	1.3269E+02	4.6310E-05	1.3268E+02	1.3271E+02	66099.5260	75.643
mean24	3.3957E+01	2.6005E-05	3.3950E+01	3.3965E+01	4633.3471	1079.15
mean25	1.3324E+02	1.9592E-05	1.3323E+02	1.3324E+02	13775.2710	362.96
mean26	3.3621E+01	1.9563E-05	3.3615E+01	3.3627E+01	75168.1397	66.517
mean27	1.3286E+02	2.7768E-05	1.3285E+02	1.3287E+02	72290.7471	69.165
mean28	3.3967E+01	1.9263E-06	3.3964E+01	3.3969E+01	7635.2819	654.85
mean29	1.3344E+02	9.1469E-06	1.3344E+02	1.3345E+02	21692.2676	230.49

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
mean30	3.3965E+01	1.7468E-06	3.3962E+01	3.3968E+01	30107.9168	166.06
mean31	1.3372E+02	2.5383E-05	1.3371E+02	1.3373E+02	29180.7359	171.34
mean32	3.3807E+01	9.2728E-06	3.3800E+01	3.3811E+01	51407.7307	97.261
mean33	1.3317E+02	5.2719E-05	1.3316E+02	1.3318E+02	64555.7808	77.452
mean34	3.3589E+01	1.6618E-05	3.3583E+01	3.3596E+01	24340.1354	205.42
mean35	1.3289E+02	1.0972E-05	1.3288E+02	1.3289E+02	25239.1536	198.10
mean36	3.3957E+01	5.4964E-06	3.3953E+01	3.3961E+01	33494.4226	149.27
mean37	1.3388E+02	1.4084E-05	1.3387E+02	1.3389E+02	19086.3787	261.96
mean38	3.3869E+01	1.1937E-04	3.3852E+01	3.3888E+01	146107.2442	34.221
mean39	1.3381E+02	2.1818E-04	1.3379E+02	1.3383E+02	126098.6469	39.651
mean40	3.3899E+01	6.3637E-06	3.3896E+01	3.3903E+01	29684.1219	168.44
mean41	1.3334E+02	1.2572E-05	1.3334E+02	1.3335E+02	28284.2642	176.77
mean42	3.4005E+01	8.7598E-05	3.3990E+01	3.4022E+01	34455.4313	145.11
mean43	1.3439E+02	7.7585E-05	1.3437E+02	1.3441E+02	44015.9496	113.59
mean44	3.4089E+01	1.0985E-05	3.4087E+01	3.4092E+01	10783.1209	463.68
mean45	1.3389E+02	2.1354E-06	1.3389E+02	1.3390E+02	16814.1550	297.36
mean46	3.4030E+01	1.4371E-05	3.4023E+01	3.4037E+01	24344.4476	205.38
mean47	1.3368E+02	6.3650E-06	1.3368E+02	1.3369E+02	26869.3262	186.08
mean48	3.3974E+01	1.8305E-05	3.3969E+01	3.3982E+01	16220.0577	308.26
mean49	1.3418E+02	4.3357E-05	1.3417E+02	1.3419E+02	33984.2140	147.12
sig0	1.0397E-02	5.1172E-06	7.3910E-03	1.4208E-02	101753.1033	49.138
sig1	1.1494E-02	2.2762E-05	6.9227E-03	1.6940E-02	89742.0891	55.715
sig2	2.2147E-02	7.5291E-06	1.7002E-02	2.6493E-02	18702.3423	267.34
sig3	2.5312E-02	1.3278E-05	2.0509E-02	3.3648E-02	16758.0436	298.36
sig4	6.7377E-02	3.4771E-05	5.8035E-02	7.7321E-02	90886.8280	55.013
sig5	1.2037E-01	8.2671E-05	1.0577E-01	1.3751E-01	79241.5240	63.098
sig6	7.6083E-02	1.2216E-05	6.9972E-02	8.3122E-02	55230.0747	90.530
sig7	8.9054E-02	1.8748E-05	8.0247E-02	9.5237E-02	33024.4230	151.40
sig8	2.3056E-02	3.7189E-06	1.8870E-02	2.6342E-02	23756.2145	210.47
sig9	2.7127E-02	3.8963E-06	2.4081E-02	3.1347E-02	28984.7066	172.50
sig10	4.4864E-02	1.5913E-05	3.8071E-02	5.1483E-02	21330.6113	234.40
sig11	2.8569E-02	1.6763E-05	2.2633E-02	3.5289E-02	79364.6376	63.000
sig12	7.3945E-02	1.4626E-05	6.6090E-02	8.0964E-02	16359.3461	305.63
sig13	6.0247E-02	4.9425E-05	5.0572E-02	7.5667E-02	45307.7279	110.35
sig14	4.7560E-02	6.7658E-05	3.7400E-02	5.9961E-02	71546.1485	69.885
sig15	4.5253E-02	5.1485E-05	3.2712E-02	5.8367E-02	122438.7684	40.836
sig16	8.1149E-02	6.4461E-05	7.3386E-02	9.7219E-02	35379.4699	141.32
sig17	9.8157E-02	1.0560E-04	8.1357E-02	1.1856E-01	65447.2037	76.397
sig18	1.7481E-01	4.9086E-04	1.4254E-01	2.1546E-01	118957.1502	42.031
sig19	3.7826E-01	2.1703E-03	2.9221E-01	4.5453E-01	98079.9965	50.978
sig20	6.3216E-02	3.0365E-05	5.2831E-02	7.1833E-02	63138.9261	79.190
sig21	8.3011E-02	3.0218E-05	7.2707E-02	9.3286E-02	67121.1605	74.492
sig22	4.2591E-02	5.3225E-06	3.8782E-02	4.7331E-02	11863.5828	421.45
sig23	7.9679E-02	1.6019E-05	7.4067E-02	8.7572E-02	30533.1187	163.75
sig24	4.0758E-02	1.3113E-05	3.5724E-02	4.5320E-02	18413.7036	271.53
sig25	3.6311E-02	5.4108E-05	2.5687E-02	4.5761E-02	319437.7767	15.652
sig26	3.3717E-02	1.8502E-05	2.7022E-02	4.1190E-02	38797.8225	128.87
sig27	3.2957E-02	1.9160E-05	2.7969E-02	4.3278E-02	85318.1770	58.604
sig28	1.3705E-02	1.3658E-06	1.1576E-02	1.5649E-02	11658.0919	428.88
sig29	1.7861E-02	5.2670E-05	1.3820E-02	2.7240E-02	8839.8629	565.61
sig30	2.3851E-02	4.9658E-06	2.1458E-02	2.6093E-02	20196.0798	247.57
sig31	6.3047E-02	1.3221E-05	5.5082E-02	6.8559E-02	20046.4462	249.42
sig32	3.3400E-02	3.7970E-06	2.9305E-02	3.6581E-02	50200.1253	99.601
sig33	8.9384E-02	1.3365E-05	8.3359E-02	9.6302E-02	40222.3843	124.30

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	mean	variance	HPD_{05}	HPD_{95}	ACT	ESS
sig34	1.6552E-02	7.3247E-05	1.2323E-02	2.3667E-02	20320.9017	246.05
sig35	1.9078E-02	9.0345E-06	1.5246E-02	2.6690E-02	18222.0431	274.39
sig36	2.9753E-02	3.2640E-06	2.6535E-02	3.3308E-02	12711.3278	393.34
sig37	3.9156E-02	9.5006E-06	3.3603E-02	4.3555E-02	15530.8432	321.94
sig38	4.3736E-02	8.1981E-05	3.3489E-02	5.9352E-02	87473.4563	57.160
sig39	5.6483E-02	7.0326E-05	4.4777E-02	7.5767E-02	44498.5518	112.36
sig40	3.2181E-02	1.0031E-05	2.9040E-02	3.4087E-02	18767.5229	266.41
sig41	5.8665E-02	7.4578E-06	5.3512E-02	6.2982E-02	8008.1789	624.36
sig42	4.3597E-02	2.8268E-05	3.5905E-02	5.3344E-02	35259.9516	141.80
sig43	4.2660E-02	1.8198E-05	3.4373E-02	5.0306E-02	31057.7121	160.99
sig44	2.0933E-02	2.7451E-05	1.7862E-02	2.2511E-02	15465.7156	323.29
sig45	2.1647E-02	1.2810E-06	1.9749E-02	2.4108E-02	18882.7570	264.79
sig46	3.4843E-02	1.1288E-05	3.0400E-02	4.1280E-02	12880.6637	388.17
sig47	2.9820E-02	4.6206E-06	2.5811E-02	3.3832E-02	14872.4660	336.19
sig48	3.8268E-02	1.7943E-05	3.2641E-02	4.4989E-02	37424.5456	133.60
sig49	7.3317E-02	4.4131E-05	6.2792E-02	8.3872E-02	37615.4550	132.92
rho0	-1.6297E-02	5.2185E-02	-4.6878E-01	3.7495E-01	26512.5997	188.58
rho1	-6.2773E-01	9.5917E-03	-7.8494E-01	-4.5777E-01	26590.6405	188.03
rho2	9.3617E-01	1.7745E-04	9.1451E-01	9.5750E-01	230515.6332	21.690
rho3	3.8187E-01	2.3841E-03	2.6518E-01	4.5914E-01	44177.0986	113.18
rho4	-4.7966E-01	4.1922E-03	-5.9941E-01	-3.5422E-01	19050.5638	262.45
rho5	1.1060E-01	1.5303E-02	-1.2180E-01	2.9967E-01	57679.1470	86.686
rho6	-4.9259E-01	2.0951E-02	-6.5592E-01	-8.0627E-02	116877.5009	42.779
rho7	-9.3090E-01	3.7978E-04	-9.5990E-01	-8.9145E-01	105535.7505	47.377
rho8	7.6369E-01	8.3119E-04	6.9430E-01	8.1311E-01	46400.7472	107.75
rho9	7.9118E-01	1.3539E-02	5.9939E-01	9.1923E-01	145895.9604	34.271
rho10	2.3060E-02	3.1655E-02	-2.6114E-01	3.6095E-01	144155.6079	34.684
rho11	-6.2912E-02	7.3775E-03	-1.9036E-01	1.2205E-01	27496.0159	181.84
rho12	-2.5315E-01	7.2828E-03	-3.9030E-01	-6.6908E-02	38868.7962	128.63
rho13	5.4412E-01	5.7701E-03	4.2760E-01	7.1766E-01	86363.9160	57.894
rho14	3.1846E-02	6.8279E-03	-1.2925E-01	1.6057E-01	39234.2156	127.43
rho15	-4.9963E-03	3.9516E-03	-1.1434E-01	1.0178E-01	42726.6201	117.02
rho16	3.6540E-01	3.1031E-03	2.8682E-01	4.8095E-01	126902.3632	39.400
rho17	-7.3498E-01	5.5361E-03	-8.5863E-01	-6.2345E-01	23100.7809	216.44
rho18	-5.7640E-01	3.0451E-03	-6.7629E-01	-4.8114E-01	29505.4889	169.46
rho19	-9.6126E-01	1.9381E-04	-9.8534E-01	-9.3612E-01	42478.2570	117.70
rho20	-4.5866E-01	1.6254E-03	-5.2164E-01	-3.5626E-01	36729.7248	136.12
rho21	6.6679E-01	6.4360E-03	5.3659E-01	8.0395E-01	25260.9133	197.93
rho22	-5.5560E-01	1.3427E-03	-6.1193E-01	-4.7605E-01	25955.2914	192.63
rho23	2.9981E-01	4.8525E-03	1.7641E-01	4.3718E-01	14442.0845	346.21
rho24	-8.1558E-01	1.4636E-03	-8.8326E-01	-7.4729E-01	51720.3490	96.673
p0	9.4841E-03	8.8157E-06	3.7773E-03	1.4609E-02	19695.7896	253.86
p1	1.3091E-03	5.0283E-07	6.7313E-04	1.9354E-03	19060.7841	262.31
p2	5.2091E-01	3.1162E-02	2.0918E-01	7.0672E-01	39399.0310	126.90
p3	6.8907E-01	1.1123E-03	6.3295E-01	7.5842E-01	122348.5804	40.866
p4	4.8204E-01	1.1005E-03	4.0973E-01	5.4387E-01	25151.2564	198.79
p5	6.1286E-01	3.2653E-03	5.2855E-01	7.4532E-01	45865.3068	109.01
p6	4.5864E-01	1.9018E-03	3.7606E-01	5.4930E-01	33871.6865	147.61
p7	6.4258E-01	3.2882E-02	3.4405E-01	1.0000E+00	257471.7527	19.419
p8	4.9533E-01	2.1998E-03	4.2800E-01	5.9699E-01	82995.0090	60.244
p9	9.6980E-01	1.4932E-03	8.9655E-01	9.9985E-01	38433.8119	130.09
p10	7.1425E-01	3.0174E-03	6.2227E-01	7.9271E-01	15718.6406	318.09
p11	7.7223E-01	9.7882E-04	7.0868E-01	8.3627E-01	45482.2677	109.93
p12	4.1091E-01	2.1401E-03	3.2501E-01	4.9506E-01	29664.4736	168.55

Continued on next page

Table E.7 – *Continued from previous page*

hmmer	**mean**	**variance**	**HPD$_{05}$**	**HPD$_{95}$**	**ACT**	**ESS**
p13	7.0933E-01	8.8617E-04	6.5631E-01	7.6445E-01	47446.1868	105.38
p14	5.4156E-01	5.2855E-03	4.2643E-01	6.7479E-01	24435.7993	204.61
p15	8.4811E-01	6.0109E-04	8.0251E-01	8.9474E-01	20822.5178	240.12
p16	6.0290E-01	8.1748E-04	5.4996E-01	6.6087E-01	22349.5548	223.71
p17	5.6851E-02	7.7910E-05	3.9254E-02	7.0971E-02	14496.7123	344.90
p18	8.3788E-01	2.4751E-03	7.5064E-01	9.5049E-01	31895.1454	156.76
p19	3.6076E-02	7.1633E-05	1.8826E-02	5.1437E-02	48798.1506	102.46
p20	7.9248E-01	1.0195E-03	7.4217E-01	8.4501E-01	19238.1132	259.90
p21	2.4458E-02	3.2593E-04	1.0902E-02	6.8324E-02	37652.1742	132.79
p22	2.1440E-01	7.4085E-03	7.4024E-02	3.8601E-01	108925.6785	45.902
p23	4.4622E-01	2.4041E-03	3.5150E-01	5.2919E-01	20916.1565	239.04
p24	4.5593E-01	1.6771E-03	3.7982E-01	5.3879E-01	129941.7537	38.478